



UNIVERSITÀ DEGLI STUDI DI CATANIA
DIPARTIMENTO DI MATEMATICA E INFORMATICA
CORSO DI LAUREA TRIENNALE IN INFORMATICA

Clizia Giorgia Manganaro

Deep Learning per il riconoscimento di targhe

RELAZIONE PROGETTO FINALE

Relatore:
Chiar.mo Prof. Sebastiano Battiato

Correlatore:
Dott. Oliver Giudice

Abstract

Una delle situazioni più frequenti che un esperto forense deve affrontare durante le indagini riguarda il riconoscimento dei caratteri delle targhe dei veicoli. Spesso, però, le immagini da analizzare presentano una bassa qualità tale da rendere impossibile individuare i singoli caratteri e l'identificazione della stessa.

Uno dei primi sistemi di riconoscimento di targhe fu sviluppato nel 1976 nel Regno Unito e successivamente tali sistemi subirono una rapida evoluzione. In particolare, negli ultimi anni con l'introduzione del Machine Learning e più precisamente del Deep Learning diversi lavori hanno dimostrato che, sebbene i caratteri non siano visivamente evidenti, è possibile recuperare informazioni utili ai fini investigativi anche nei casi in cui l'immagine sia fortemente degradata.

È stato effettuato uno studio su diversi approcci per il riconoscimento automatico di targhe che utilizzano reti neurali profonde e successivamente si è ritenuto opportuno introdurre due nuovi casi che simulassero situazioni reali in cui l'immagine risulti di difficile leggibilità a causa di una forte distorsione prospettica, rumore impulsivo e sfocatura causata da movimento. L'approccio proposto si basa su algoritmi di Deep Learning con l'utilizzo di una rete neurale CNN la quale richiede una gran mole di dati. Pertanto, sono stati prodotti due *dataset* per addestrare la rete con l'obiettivo di produrre un sistema che aiuti l'esperto forense nel prosieguo delle sue indagini.

Indice

1	Introduzione	7
1.1	Evoluzione dei sistemi di riconoscimento automatico.....	9
1.2	Tasks Generali	10
1.3	Image Enhancement e Image Restoration ^[1]	11
1.4	Casi di studio basati sull'apprendimento.....	19
2	Ricostruzione forense di targhe gravemente degradate^[2]	20
2.1	Caratteristiche principali	20
2.2	Dataset	21
2.2.1	Dataset sintetico	22
2.2.2	Dataset reale.....	23
2.3	Architettura CNN	24
2.4	Risultati ottenuti	26
2.5	Conclusioni.....	28
3	Imparare a decifrare le targhe in immagini fortemente degradate^[3].....	29
3.1	Caratteristiche principali	29
3.2	Dataset	30
3.3	Architettura CNN	32
3.4	Risultati ottenuti	34
3.5	Conclusioni.....	40
4	Rete Neurale di Denoising e di Lettura di targhe degradate^[4].....	41
4.1	Caratteristiche principali	41
4.2	Dataset.....	42
4.3	Architettura rete	45
4.4	Risultati ottenuti	48
4.5	Conclusioni.....	52
5	Sperimentazioni su dati di targhe italiane	53
5.1	Dati e impostazioni sperimentali.....	53

5.2	Risultati sperimentali.....	57
5.3	Conclusioni e sviluppi futuri.....	63
6	Ringraziamenti.....	65
7	Riferimenti.....	67

Capitolo 1

Introduzione

Il riconoscimento delle targhe automobilistiche rappresenta in ambito forense una delle tematiche più frequenti che gli esperti del settore devono affrontare ogni giorno durante le loro indagini.

L'incremento del numero di veicoli ha prodotto nel corso degli anni un aumento delle violazioni delle norme del codice della strada che ha portato alla diffusione in tutto il mondo di sistemi di localizzazione e riconoscimento delle targhe, ognuno con funzionalità abbastanza differenti: controllo della velocità, la gestione dei parcheggi, il monitoraggio di aree a traffico limitato, la ricerca di veicoli rubati ecc...

Tuttavia, un problema che viene frequentemente riscontrato dai sistemi di riconoscimento automatico delle targhe consiste nella poca leggibilità dei singoli caratteri causando non poche problematiche nel prosieguo delle indagini forensi.

Un esempio purtroppo ancora molto frequente riguarda le immagini acquisite da dispositivi di videosorveglianza, che, seppur registrando l'informazione nel suo insieme, una buona percentuale delle immagini catturate risultano inutilizzabili.

Difatti, spesso i frames delle immagini vengono acquisiti da sistemi di videosorveglianza con bassa qualità, causata da scarsa illuminazione, sfocatura di movimento, rumore del sensore o altre numerose cause artificiali o naturali.

Inoltre, le telecamere sono in genere configurate per massimizzare il campo visivo limitando ulteriormente la risoluzione effettiva degli oggetti nel video.

Un'ulteriore causa di perdita intrinseca di informazioni potrebbe essere causata da una forte compressione di immagini e video.

La disciplina che si occupa di tutte le attività di analisi delle immagini e dei video volte ad estrapolare dati e informazioni ad uso forense, prende il nome di *Image/Video Forensics* che appartiene ad una specifica area della *Digital Forensics*.

Infatti, l'esperto forense viene chiamato in causa per effettuare miglioramenti del dato in sé e per analizzare ed interpretare il contenuto nelle situazioni più disparate e intricate.

In particolar modo l'obiettivo finale dello studio svolto, consiste nell'analizzare i vari approcci utilizzati dall'esperto forense per il riconoscimento di targhe in situazioni in cui, come già accennato, l'identificazione dei caratteri risulta difficoltosa.

Nei casi di immagini fortemente degradate si reputa di fondamentale importanza, l'introduzione di metodi computazionali con l'obiettivo di ricostruire parzialmente o completamente una targa.

Negli ultimi anni un grande aiuto nella risoluzione delle problematiche riscontrate viene dato dal *machine learning*, in particolare di una sua sottocategoria ovvero il *deep learning*, grazie all'introduzione di reti neurali profonde che hanno ottenuto straordinari risultati nella decifrazione dei singoli caratteri delle targhe.



Figura 1.1 – Esempio di targa fortemente degradata

1.1 Evoluzione dei sistemi di riconoscimento automatico

Uno dei primi sistemi di riconoscimento di targhe fu sviluppato nel 1976 nel Regno Unito quando il dipartimento di sviluppo scientifico della *Police Scientific Development Branch - PSDB* (adesso conosciuta come *Home Office Scientific Development Branch*) presentò l'*Automatic Number Plate Recognition (ANPR)*[7].

Ad oggi vengono utilizzati diverse denominazioni per fare riferimento ai sistemi di riconoscimento:

- Automatic License Plate Recognition - ALPR
- Automatic Vehicle Identification - AVI
- Car Plate Recognition - CPR
- Mobile License Plate Reader – MLPR

I primi prototipi effettuavano letture di bassa precisione e funzionavano solo in condizioni restrittive che rendevano l'applicazione nel mondo reale quasi impraticabile. Essi utilizzavano il riconoscimento ottico dei caratteri (OCR), fotocamere e processori di dati.

In generale, i primi sistemi venivano utilizzati nei veicoli della polizia tramite una telecamera che fotografava e confrontava fino a 3.000 targhe all'ora con un database di "targhe conosciute" per consentire all'agente di polizia di identificare qualsiasi veicolo che entra nel raggio della telecamera. Se una targa fotografata dalla fotocamera corrisponde una targa del database avveniva un "hit".

La tecnologia ANPR si è successivamente evoluta e adattata nei tempi, trovando nuovi sbocchi che non si limitano alle applicazioni negli ambiti della sicurezza e delle forze dell'ordine. Nel 2006 infatti, viene implementato il primo sistema di telecamere statiche per la gestione dei parcheggi: *bayGUARDIAN* che consentiva agli operatori dei parcheggi di monitorare i veicoli in entrata e in uscita dai parcheggi calcolando il tempo trascorso dal veicolo all'interno del parcheggio.

Nel 2009 viene progettato il primo ANPR mobile nominato *streetSWEEPER*, per indagini sul traffico nelle strade, sorveglianza mobile e controllo di veicoli non tassati.

È possibile, dunque, affermare che dopo diversi decenni dall'uscita del primo prototipo, le limitazioni dovute alla velocità del veicolo, alle fluttuazioni della luce, all'inclinazione angolare, alla segmentazione dei caratteri e al conseguente riconoscimento delle targhe dei veicoli, sono state risolte con la tecnologia degli algoritmi odierni.

1.2 Tasks Generali

Il riconoscimento automatico delle targhe è un argomento di ricerca che viene studiato in tutto il mondo. In un contesto generalizzato è possibile suddividere gli approcci utilizzati in due iper-categorie:

- approcci basati sul riconoscimento;
- approcci basati sull'apprendimento (di recente successo).

I vari approcci utilizzati implicano l'acquisizione di video fotografici o immagini di targhe dalle quali vengono successivamente elaborati una serie di algoritmi in grado di fornire una conversione alfanumerica delle immagini della targa acquisita in una voce di testo per il riconoscimento dei singoli caratteri.

È possibile effettuare una suddivisione delle fasi svolte dai vari algoritmi per il riconoscimento automatico della targa in quattro fasi principali ognuno con la funzione di svolgere un task differente.

1. *Acquisizione dell'immagine*
2. *Estrazione della targa dall'immagine*

Una volta acquisita l'immagine è necessario individuare/localizzare la targa nell'immagine, ritagliarla per effettuare le dovute operazioni su di essa.

3. *Segmentazione dei caratteri*

In questa fase viene effettuata un partizionamento dell'immagine dividendo le aree dei singoli caratteri per semplificarne l'analisi.

4. *Riconoscimento dei caratteri*

In questa fase viene effettuata una etichettatura dei singoli caratteri per poter procedere al riconoscimento di quest'ultimi. A tal fine può essere eseguita su un modello di corrispondenza che avrà risultati ottimali nei casi in cui le immagini sono non-ruotate ed a dimensioni fisse.

Un ulteriore approccio potrebbe essere l'utilizzo di un modello di corrispondenza di caratteristiche.

La performance di ogni fase si basa sulla robustezza della fase precedente.

1.3 Image Enhancement e Image Restoration^[1]

Come già anticipato, le immagini a bassa risoluzione ostacolano la capacità di isolare i singoli caratteri, rendendo difficile l'estrazione e l'identificazione dei caratteri. Inoltre, le immagini digitali presentano sempre determinati difetti che per vari motivi possono limitare l'utilizzabilità delle prove investigative. Per questo motivo vengono apportati dei miglioramenti della qualità dell'informazioni al fine di ricreare l'immagine quanto più priva di difetti.

Una volta identificati i difetti, tramite opportuni algoritmi vengono effettuate tecniche di miglioramento dell'immagine (*Image Enhancement*) e di restauro della stessa (*Image Restoration*).

Le tecniche di miglioramento sono processi di regolazione delle immagini digitali che consentono di esaltare le caratteristiche dell'immagine per poter

successivamente effettuare l'estrazione di informazione (*Feature Detection*), nel nostro caso per evidenziare i caratteri presenti nelle targhe.

Le tecniche di restauro sono procedimenti/algoritmi con i quali è possibile ricostruire il modello matematico che approssima il difetto che genera un determinato disturbo. Il ripristino dell'immagine viene eseguito invertendo il processo che ha causato il difetto/disturbo dell'immagine per potere ottenere un ripristino di qualità dell'informazione.

È necessario tenere in considerazione che le tecniche di elaborazione delle immagini sono in generale rivolte all'ottenimento dei seguenti obiettivi: miglioramento di qualità, ripristino di qualità o restauro ed infine estrazione di informazione. In aggiunta, si ritiene opportuno precisare che l'operazione di miglioramento di un dettaglio può portare ad un peggioramento della qualità complessiva dell'immagine.

Le operazioni che l'esperto forense esegue nell'analisi di indagine possono essere riassunte nelle seguenti fasi:

1. Acquisizione del filmato, conversione del filmato selezione dei fotogrammi contenenti la targa di interesse
2. Deinterlacciamento
3. Correzione degli artefatti causati dalla compressione
4. Correzione disturbi analogici
5. Riduzione del rumore
6. Definizione dei dettagli
7. Correzione della sfocatura e miglioramento dei dettagli
8. Correzioni geometriche
9. Regolazioni di intensità e colori
10. Stabilizzazione

Analizzeremo nel dettaglio alcune di queste fasi.

Uno dei difetti presenti nei filmati provenienti da sistemi con dispositivi analogici è dovuto alla presenza di segnale interlacciato. L'interlacciamento è una tecnica di "compressione" utilizzata nei sistemi di acquisizione analogica per ridurre la banda del segnale trasmessa. L'immagine sarà formata da linee pari e dispari appartenenti a sequenze temporali differenti, pertanto, il difetto è per la maggior parte dei casi visibile a vista d'occhio soprattutto in immagini con auto in movimento. Per poter effettuare deinterlacciamento e difatti, eliminare il difetto verrà fatta l'interpolazione delle linee per ricreare le linee mancanti.



Figura 1.2 – Fotogramma interlacciato (sinistra), risultato del deinterlacciamento (destra)

Come vedremo in seguito, uno dei problemi principali nella leggibilità di una targa è generato dalla compressione della codifica di immagini e filmati, che causa una perdita considerevole di dettagli e la creazione di artefatti. Una risoluzione in questi contesti consiste nell'utilizzare algoritmi di *deblocking*.

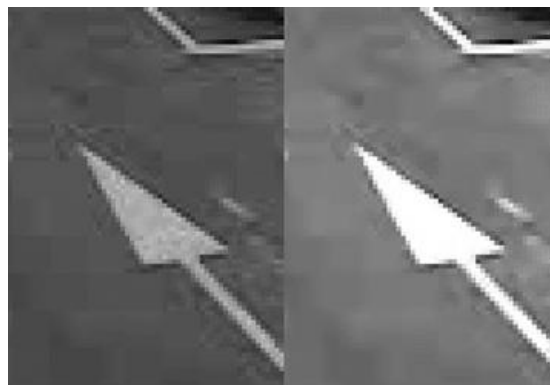


Figura 1.3 – Artefatto da compressione (sinistra), riduzione della blocchettatura (destra)

Nel prosieguo della stessa analizzeremo una soluzione adottata da Kaiser et al. [3] per le immagini fortemente degradate a causa della compressione. Tale approccio sarà basato sull'apprendimento, usufruendo di reti neurali al fine di identificare la migliore modalità di riconoscimento caratteri di targa in immagini molto compresse con elevata perdita di dati.

Per quanto riguarda il rumore, le tipologie a cui si fa più comunemente riferimento sono di due tipi: il rumore impulsivo ed il rumore gaussiano bianco. I filtri più comuni per correggere rispettivamente disturbi di questo tipo sono il filtro mediano e media (o N-box). Il filtro mediano è molto efficace nei rumori impulsivi poiché elimina i valori estremi nell'immagine, mentre il filtro media risulta maggiormente efficace nel rumore gaussiano bianco.

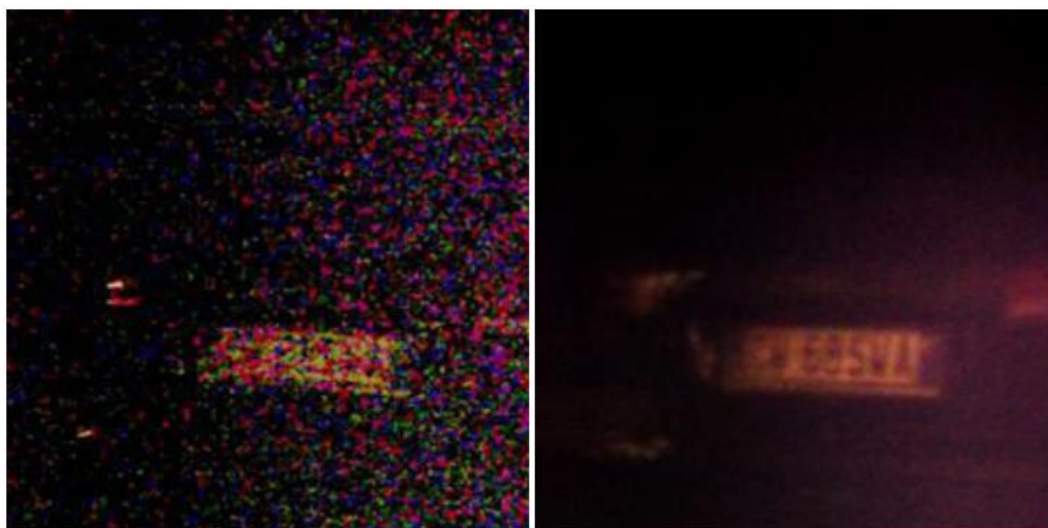


Figura 1.4 - Immagine con rumore (sinistra), immagine migliorata (destra)

La definizione dei dettagli risulta di fondamentale aiuto nella leggibilità della targa. Tale operazione viene anche chiamata *sharpening* ed è un'operazione che consiste nel migliorare il contrasto locale dell'immagine incrementando la nitidezza.

Spesso l'esperto forense dovrà analizzare immagini fortemente sfocate. Nella risoluzione di questa tipologia di difetti risulta conveniente operare nel dominio delle frequenze.

Infatti, un'immagine sfocata può essere vista come il risultato di un procedimento matematico chiamato "convoluzione" di una immagine ideale "nitida" con la *PSF* - *Point Spread Function* anche detta funzione di *blurring*. Facendo una stima corretta della *Point Spread Function* è possibile invertire il modello e quindi ricostruire l'immagine "nitida" applicando il processo inverso. Semplificando, viene effettuata una "deconvoluzione" dell'immagine sfocata con la *Point Spread Function*. Una delle tecniche più utilizzate per effettuare la deconvoluzione è quella di Wiener.

Nelle immagini di seguito abbiamo degli esempi in cui viene effettuata la deconvoluzione utilizzando nella figura 1.4 come modello della *Point Spread Function* un cerchio e nella figura 1.5 una linea.



Figura 1.5- Immagine sfocata (sinistra) correzione sfocatura ottica (destra)



Figura 1.6 - Immagine sfocata (sinistra) correzione sfocatura ottica (destra)

Spesso le immagini vengono catturate da angolazioni che non ne consentono la corretta identificazione della targa, per questo motivo è necessario effettuare correzioni geometriche sui pixel, al fine di "spostare" gli stessi all'interno dell'immagine modificandone le coordinate. Esempi di algoritmi che effettuano

correzioni geometriche sono il ritaglio, la riflessione, la rotazione dell'immagine o la correzione prospettica.



Figura 1.7 – Immagine originale (sinistra) ritaglio dell'immagine (destra)



Figura 1.8 – Immagine originale (sinistra) correzione prospettica (destra)

Un ulteriore step che l'esperto forense effettua durante la sua indagine e in particolare nella fase di *image enhancement* consiste nella regolazione della luminosità e dei colori. Un esempio molto frequente consiste nell'analisi di immagini in riprese notturne. In questi casi potrebbe capitare che la targa automobilistica rifletta la luce saturando il sensore o che sia completamente al buio. Vengono quindi modificati i valori dei singoli pixel. Le operazioni più utilizzate per la

regolazione dell'intensità sono: esposizione, luminosità e contrasto, curve, livelli, mixer canali, equalizzazione dell'istogramma e bilanciamento del bianco.



Figura 1.9 – Equalizzazione dell'istogramma



Figura 1.10 – Regolazione di intensità

Quando il sistema cattura le targhe di auto che sono in movimento, come ad esempio nei casi di ANPR posti in autostrada, sarà necessario effettuare una stabilizzazione dell'immagine.



Figura 1.11 – Stabilizzazione

Infine, ottimi risultati sono ottenuti integrando vari fotogrammi delle riprese contenente l'informazione e applicando vari algoritmi per ottenere un'immagine con qualità maggiore.

Tra questi troviamo: la media dei fotogrammi, super-risoluzione e pseudo-super-risoluzione.

- La media dei fotogrammi crea un'immagine in cui ogni pixel è la media di tutti i pixel nella stessa posizione dei fotogrammi considerati.
- La super-risoluzione effettua in un primo momento l'allineamento dei fotogrammi facendo interpolazione dei pixel e successivamente effettua la media.
- Pseudo-super-risoluzione effettua una stabilizzazione dell'immagine e successivamente viene effettuata la media. L'immagine apparirà con meno artefatti rispetto alla super-risoluzione che applica algoritmi di interpolazione.



Figura 1.12 – Originale(in alto a sx), media(in alto a dx), super-risoluzione (in basso a sx), pseudo-super-risoluzione (in basso a dx).

1.4 Casi di studio basati sull'apprendimento

Negli studi più recenti, con approcci basati sull'apprendimento, si è scoperto che quando si tratta di immagini a bassa risoluzione con notevoli difetti risulta efficace combinare le fasi di segmentazione e riconoscimento dei caratteri ed in particolare, introdurre il mondo delle Reti Neurali per svolgere alcuni task sopraelencati.

Nel dettaglio, con il recente successo delle Reti Neurali Convoluzionali (CNN), sono emersi una molteplicità di metodi di riconoscimento che possiedono una maggiore precisione ed accuratezza nel riconoscimento di caratteri in targhe fortemente degradate, superando di gran lunga i risultati ottenuti dai tradizionali sistemi ANPR.

Di seguito verranno analizzati tre casi di studio di recente successo con applicazioni, finalità ed architetture CNN differenti.

1. Forensic Reconstruction of Severely Degraded License Plates by Lorch et al. [2]
2. Learning to Decipher License Plates in Severely Degraded Images by Kaiser et al. [3]
3. Neural Network for Denoising and Reading Degraded License Plates by Rossi et al. [4].

Capitolo 2

Ricostruzione forense di targhe gravemente degradate^[2]

Il primo studio che andremo ad analizzare è il lavoro di ricerca svolto da Lorch et al.[2] che pone come obiettivo il recupero delle informazioni di targhe automobilistiche statunitensi da immagini a bassa risoluzione e fortemente degradate.

Spesso anche con l'utilizzo di tecnologie avanzate di riduzione del rumore e super-risoluzione viste nel paragrafo 1.4 non si riesce ad estrarre informazioni utili in immagini con bassa qualità.

A tal fine viene mostrato un approccio per il riconoscimento di caratteri che si basa sull'implementazione di reti neurali convolutive (*Convolutional Neural Network - CNN*) per l'analisi di immagini e la decifrazione del contenuto.

Le CNN sono una classe di reti neurali profonde che permettono di applicare le reti neurali in maniera efficiente al processamento di immagini. Sono composte da una gerarchia di livelli dove il livello di input è formato dalle singole immagini con uno o più canali, come immagini a scala di grigi o RGB.

2.1 Caratteristiche principali

Il metodo implementato da Lorch et al. [2] prende in considerazione targhe contenenti da cinque a sette caratteri e non pone limiti sul formato dei caratteri presenti o sulle loro posizioni.

Tuttavia, viene ipotizzato che l'esperto forense abbia effettuato le dovute correzioni di *image enhancement* in particolare per quanto riguarda la distorsione prospettica ed il ritaglio dell'immagine effettuato correttamente ai bordi della targa.

La valutazione dell'efficacia del metodo è stata effettuata su immagini estremamente degradate con una risoluzione compresa tra 12×6 e 55×27.5 pixel di larghezza e rapporto di segnale-rumore (*signal-to-noise ratio* – *SNR*) che va da -3.0 a 20.0 dB.

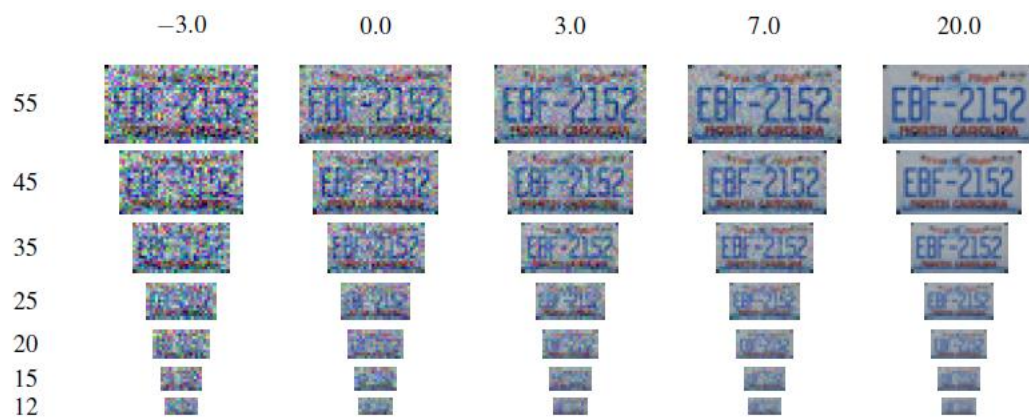


Figura 2.1 – Sull'asse x la larghezza dell'immagine in pixel e sull'asse y SNR in dB

2.2 Dataset

Per poter addestrare la CNN sono stati generati un dataset sintetico e un dataset contenente immagini reali.

Una volta acquisito il dataset per simulare immagini a bassa risoluzione e qualità entrambi i dataset sono stati sottoposti a vari livelli di degradazione. Ogni immagine è stata sottocampionata di una larghezza di 400 pixel con una larghezza compresa da 10 a 60 a pixel. Inoltre, è stato aggiunto il rumore gaussiano a media zero che produce immagini con un SNR che varia da -3:0 a 20:0 dB come visto in figura 2.1.

2.2.1 Dataset sintetico

Il dataset sintetico è stato creato effettuando le misurazioni del Font e la posizione dei caratteri da 259 targhe reali che contengono da cinque a sette caratteri. Le misure effettuate includono la larghezza e l'altezza del carattere, la distanza del carattere dall'angolo superiore sinistro della targa e la posizione e larghezza degli spazi tra i caratteri.

Le stringhe alfanumeriche sono state generate in maniera randomica e sono state renderizzate con dimensioni di caratteri e spaziatura tratte dalle misurazioni delle targhe reali. Il font è stato scelto utilizzando uno dei quattro font utilizzati nelle targhe reali. La luminosità è stata selezionata in maniera casuale da nero a bianco con uno sfondo randomico in scala di grigi per creare contrasto. Le immagini possiedono una risoluzione di 400×200 pixel.



Figura 2.2 – Targhe sintetiche

In un primo momento erano state generate immagini a scala di grigi come in alto in figura 2.2, successivamente per aumentare la precisione del sistema sono state generate una seconda serie di immagini che cercavano di avvicinarsi maggiormente alla realtà come è possibile vedere in basso in figura 2.2.

Per questo motivo sono state aggiunte le seguenti caratteristiche:

- inserimento di oggetti generati casualmente estratti da *Hemera Photo Object*;
- inserimento dei caratteri in rilievo per aggiungere ombreggiature ai bordi e rendere le targhe più verosimili possibile;
- aggiunta di un frame all'immagine;
- immagini a colori estratti in maniera casuale da colori di targhe del mondo reale;
- sfondo, caratteri e frame sono stati scelti in maniera randomica da immagini scaricate da Flickr.

2.2.2 Dataset reale

Per generare il dataset contenente immagini reali sono state scaricate 6.196 immagini da *plateshake.com*, un sito che contiene una grande collezione di targhe. Ogni immagine è stata esaminata singolarmente, ritagliata ed eliminata la distorsione prospettica.

Sono state mantenute solo le targhe con cinque o sette caratteri rimuovendo quelle con bassa qualità.

Per integrare il dataset sono state aggiunte 1.771 fotografie di targhe reali acquisite in diversi stati degli USA, anche ad esse sono state applicate operazioni di *image enhancement* riguardanti il ritaglio e distorsione prospettica. Di seguito è possibile visionare un esempio delle targhe reali.



Figura 2.3 – Targhe reali

2.3 Architettura CNN

L'architettura della rete neurale CNN proposta da Lorch et al. [2] è formata da otto *layers* convoluzionali, cinque di *max-pooling*, due livelli completamente connessi, l'ultimo strato completamente connesso genera sette livelli completamente connessi ciascuno dei quali è seguito da un layer *soft-max* di output.

I *layers* convoluzionali utilizzano un kernel di dimensione 3×3 e padding di un pixel mentre i numeri dei filtri del kernel sono $\{64, 64, 128, 128, 256, 256, 512, 512\}$.

Viene inserito il *max-pooling* dopo il secondo, il quarto, il sesto, il settimo e ottavo livello e ogni strato di *max-pooling* dispari riduce la dimensione spaziale di due. L'ultimo livello di pooling è seguito da due livelli *fully-connected* costituiti rispettivamente da 1024 e 2048 unità.

Il numero dei filtri è indicato in ciascuna casella di seguito riportata (Figura 2.4).

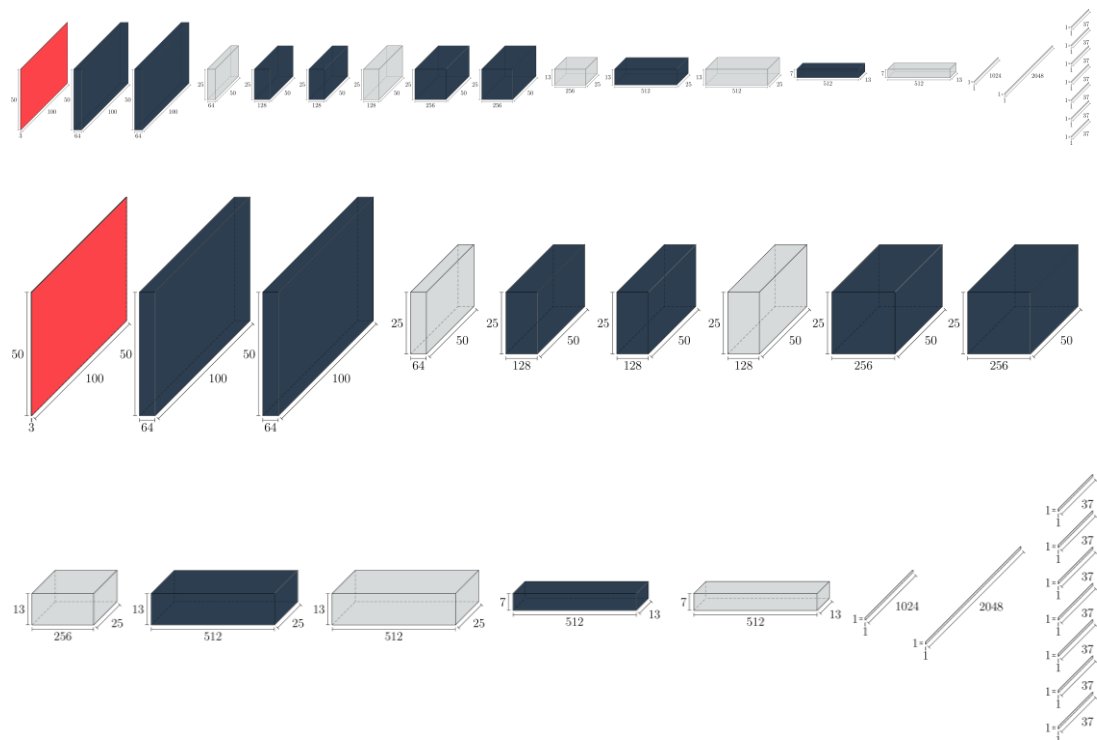


Figura 2.4 – Architettura CNN

Tutti i livelli convoluzionali, compresi i primi due *fully-connected*, usano la funzione di attivazione ReLU. Il dropout viene applicato dopo la funzione di attivazione dei primi due strati completamente connessi e la probabilità di dropout è 0.5.

Per tutti i livelli convoluzionali e i sette livelli di output viene effettuata l'inizializzazione del peso "*Xavier*". Difatti, i primi due livelli *fully-connected* sono inizializzati usando una distribuzione normale con media zero e deviazione standard pari a 0.005.

Per quanto riguarda il bias nei livelli convoluzionali i primi due livelli completamente connessi e i sette livelli di output sono inizializzati con un valore fisso di 0.1, 0.1 e 0.

La CNN è stata implementata utilizzando TensorFlow v.1.4 ed è stata utilizzata una GeForce GTX 1080 Ti GPU.

Come anticipato, la rete avrà sette livelli di output dove ciascun livello corrisponde ad un singolo carattere. Per questo motivo questa architettura ha il limite di poter leggere solamente una targa che possiede sette caratteri.

Ogni livello di output è costituito da 37 unità, dove 26 sono le possibili lettere, 10 possibili cifre e un carattere speciale *null* "◇" introdotto per consentire la lunghezza variabile delle targhe, ad esempio una targa di cinque caratteri del tipo "ABC12" sarà rappresentata da "ABC12◇◇".

Pertanto, la predizione di ogni livello può essere interpretata come la distribuzione di probabilità discreta su 37 possibili caratteri.

Per valutare l'accuratezza dei singoli caratteri viene introdotta una metrica: viene assegnato "Top-1" ai casi in cui il carattere predetto è il più probabile mentre "Top-5" nei casi in cui il carattere si è classificato tra i cinque più probabili. Inoltre, viene calcolata la distanza di Levenshtein, ovvero data due stringhe tale distanza calcola il numero minimo di caratteri da inserire, eliminare o sostituire per trasformare una stringa in un'altra.

2.4 Risultati ottenuti

Analizzeremo i risultati ottenuti sulla base di tre esperimenti svolti:

- 1- Synthetic-I
- 2- Synthetic-II
- 3- Real-world

	width (pixels)	(a) Top-1					(b) Top-5					(c) Levenshtein distance				
		SNR (dB)					SNR (dB)					SNR (dB)				
		-3.0	0.0	3.0	7.0	20.0	-3.0	0.0	3.0	7.0	20.0	-3.0	0.0	3.0	7.0	20.0
Synthetic-I	55	71.3	80.1	84.0	85.7	85.8	91.0	95.1	96.2	96.4	96.1	1.8	1.2	1.0	0.8	0.8
	45	60.8	74.8	80.6	83.3	85.0	85.0	92.3	95.1	95.6	95.9	2.5	1.6	1.2	1.0	0.9
	35	43.8	57.8	68.1	74.8	79.0	71.5	83.1	89.3	92.0	93.4	3.7	2.7	2.0	1.6	1.3
	25	21.3	30.9	39.8	50.3	59.9	46.0	57.5	67.5	76.5	83.6	5.3	4.7	4.0	3.3	2.5
	20	15.2	19.1	24.6	31.3	40.6	35.3	43.6	51.3	59.4	69.4	5.8	5.5	5.1	4.6	3.9
	15	11.1	12.6	14.3	16.5	19.9	26.9	29.9	34.5	38.8	44.3	6.1	6.0	5.9	5.7	5.5
	12	10.5	11.3	11.9	13.0	14.4	23.9	26.2	28.2	31.3	35.1	6.2	6.1	6.0	6.0	5.9
Synthetic-II	55	76.8	83.1	85.6	87.1	88.1	93.8	96.0	97.0	97.4	97.5	1.5	1.0	0.8	0.7	0.7
	45	69.6	80.1	84.8	86.9	88.1	90.2	95.1	96.6	97.1	97.6	2.0	1.2	0.9	0.8	0.7
	35	53.8	68.3	77.2	83.5	86.5	80.5	89.8	94.2	95.9	96.7	3.1	2.1	1.5	1.0	0.8
	25	29.3	42.5	55.1	66.9	77.8	56.7	70.1	80.9	88.5	93.7	4.8	3.9	3.0	2.2	1.4
	20	19.6	27.3	37.3	47.0	60.5	43.6	55.0	65.7	74.9	84.6	5.5	5.0	4.3	3.6	2.6
	15	12.5	15.2	19.1	22.9	31.4	30.1	36.2	42.4	48.9	58.4	6.0	5.8	5.6	5.3	4.7
	12	10.5	12.2	13.6	16.3	19.9	26.9	30.6	33.2	39.4	46.3	6.2	6.1	6.0	5.8	5.5
Real-world	55	89.2	93.9	95.5	96.0	96.2	98.4	99.3	99.6	99.6	99.7	0.7	0.4	0.3	0.2	0.2
	45	82.1	91.8	94.9	95.8	96.3	96.5	98.9	99.5	99.6	99.7	1.2	0.5	0.3	0.3	0.2
	35	67.8	82.7	90.9	94.3	95.6	90.6	96.7	98.9	99.4	99.6	2.2	1.1	0.6	0.3	0.3
	25	40.3	57.8	72.9	84.7	92.2	73.0	85.9	93.3	97.3	99.1	4.1	2.8	1.8	1.0	0.5
	20	26.4	38.8	52.5	65.9	81.8	60.3	72.0	83.3	90.8	96.5	5.0	4.1	3.2	2.3	1.2
	15	16.7	21.5	27.9	38.2	52.9	46.7	53.8	62.2	72.2	83.2	5.6	5.3	4.9	4.2	3.2
	12	13.5	15.9	18.8	24.3	32.9	41.0	45.8	51.0	58.4	68.8	5.8	5.6	5.4	5.1	4.6

Figura 2.5 – Risultati ottenuti

Il primo esperimento (Synthetic-I) è stato effettuato addestrando la rete su 10 milioni di immagini in scala di grigi con 2 mila immagini usate per la validazione. La rete è stata testata su mille immagini reali sempre convertite in scala di grigi.

Nella parte superiore della tabella in figura 2.5 sono mostrati i risultati ottenuti con il primo esperimento. Pertanto, viene rappresentata l'accuratezza in percentuale rispettivamente in Top-1, Top-2 e distanza di Levenshtein in funzione del SNR e della risoluzione dell'immagine.

È possibile notare come l'accuratezza diminuisce con una bassa risoluzione e con l'aumentare del rumore. Il calo più repentino della precisione si verifica per la risoluzione inferiore a 25 pixel (o circa 3 pixel per carattere per una targa a sette caratteri).

Il secondo esperimento (Synthetic-II) è stato svolto per risolvere il divario tra immagini generate sinteticamente e le immagini di test del mondo reale.

A tal fine è stata addestrata nuovamente la rete su più immagini sintetiche realistiche e a colori. Sono state generate 10 milioni di immagini di training con 2 mila immagini usate per la validazione, al centro in figura 2.5 è possibile verificare l'accuratezza ottenuta in particolare l'accuratezza risulta migliorata in quasi tutte le risoluzioni e livelli di rumore.

Nell'ultimo esperimento svolto (Real-World) la CNN è stata addestrata con immagini del mondo reale scaricate da *plateshake.com* e prendendone una randomicamente l'immagine è stata degradata con livelli di risoluzione e rumore sopradetti.

Sono state ottenute 1 milioni di immagini degradate per fare un training della rete. Un *validation set* di 10 mila immagini è stato creato dalle 771 targhe rimanenti dal mondo reale per la collezione non utilizzata per il testing. Queste immagini di *training/validation* sono state usate per effettuare *fine-tune* della rete con un ridotto tasso di apprendimento di 0.0001.

Il *fine-tuning*, com'è possibile osservare in figura 2.5 in basso, migliora l'accuratezza rispetto alle soluzioni precedenti per tutte le risoluzioni e livelli di rumore.

In conclusione, in figura 2.6 viene rappresentata la frequenza con cui ogni carattere alfanumerico viene confuso o riconosciuto con altri.

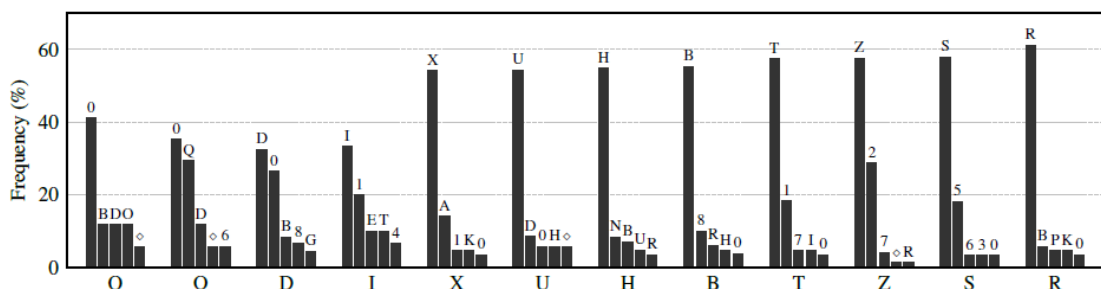


Figura 2.6 – Frequenza(%) di errato riconoscimento per carattere

2.5 Conclusioni

Sebbene nelle immagini altamente degradate delle targhe non vi siano informazioni visivamente evidenti, è stato ad ogni modo possibile estrarre caratteristiche distintive dalle stesse .

Tale studio dimostra dunque come con un dataset di training appropriato, una rete neurale convolutiva (*Convolutional Neural Network - CNN*) possa decifrare accuratamente anche le immagini più degradate in modo significativo superando i tassi di riconoscimento umano.

In conclusione, è possibile effettuare miglioramenti con un dataset fortemente diversificato che porterà ad una maggiore flessibilità e tasso di riconoscimento ancora più elevato.

Capitolo 3

Imparare a decifrare le targhe in immagini fortemente degradate^[3]

Il lavoro svolto da Kaiser et al. [3] pone l'obiettivo di introdurre un'analisi accurata del sistema di riconoscimento delle targhe proposto da Lorch et al.[2] su immagini con compressione JPEG.

Difatti, l'acquisizione delle immagini da telecamere di videosorveglianza è soggetta a una forte compressione che causano la degradazione dell'immagine più diffusa nelle indagini forensi.

Tale approccio mira a colmare questa lacuna ed analizzare l'impatto della compressione JPEG sul riconoscimento automatico delle targhe su immagini fortemente degradate.

3.1 Caratteristiche principali

La compressione delle immagini porta ad una perdita intrinseca di informazioni complicando ulteriormente le indagini nelle immagini con risoluzione molto bassa rendendo illeggibile la targa all'occhio umano.

Dunque, l'utilizzo della compressione di tipo *lossy* presenti in quasi tutti i sistemi di acquisizione video ha un forte impatto sulle indagini forensi.

A tal fine, è stato sviluppato un approccio basato sul lavoro svolto da Lorch et al.[2] integrando una soluzione atta a risolvere la questione della degradazione delle immagini causata dalla compressione.

Dunque, è stata testata l'efficacia di una rete neurale convolutiva – CNN su un dataset formato da immagini reali contenente targhe di nazionalità ceca.

Utilizzando per la fase di training solamente un dataset formato da targhe generate sinteticamente con una width maggiore di 30 pixels, rapporto segnale-rumore (*Signal-to-Noise Ratio*) sopra i -3 dB e fattore di qualità JPEG fino a 15 è stato possibile ricostruire almeno parzialmente la targa.

3.2 Dataset

Per ottenere un alto tasso di previsione, la rete necessita di essere addestrata con un grande numero di esempi durante la fase di *training*.

Analogamente a quanto già sviluppato da Lorch et al. [2] è stato realizzato un dataset sintetico che comprende 10 milioni di immagini di targhe ceche per il training, 2 000 per la validazione e 750 000 per il testing.

Le targhe ceche sono formate da sette caratteri con posizioni fisse. Per di più, le normative della Repubblica Ceca specificano il font, la dimensione e le misurazioni come spaziatura e l'offset. Sono state prese in considerazione esclusivamente le targhe che hanno nella prima posizione solamente cifre da 1 a 9 (sebbene ci sia una bassissima probabilità che nella prima posizione sia presente un carattere non-numerico) e nella seconda posizione viene indicata con un carattere la regione dove viene rilasciata la targa. Le possibili lettere che rappresentano le regioni sono A, B, C, E, H, J, K, L, M, P, S, T, U, Z. La terza posizione non ha nessun vincolo mentre nelle posizioni da quattro a sette sono presenti solamente cifre da 0 a 9. Inoltre, dal momento in cui i caratteri G, O, Q e W non vengono utilizzate in tutte le targhe, non verranno inserite nel *training set*.



Figura 3.1 – Esempio di targa ceca

In figura 3.2 viene rappresentato il processo per generare le targhe. Quest'ultimo è costituito da cinque fasi:

1. viene generato frame e font;
2. la targa viene ritagliata ;
3. viene effettuato il sottocampionamento per degradare l'immagine;
4. viene aggiunto il rumore all'immagine;
5. infine, l'immagine viene compressa.



Figura 3.2 – Processo di generazione targhe sintetiche

Per poter effettivamente creare dell'immagini sintetiche quanto più verosimili possibile viene effettuato il sottocampionamento dell'immagine della targa con larghezza da 30 a 180 pixels con aspect ratio fisso. Per il training sono stati considerati sette livelli di risoluzione: 20, 50, 70, 90, 120, 150 e 180 pixels. Per ottenere una dimensione uniforme come input per la rete le immagini vengono sovracampionate con una dimensione di 180×44 pixels effettuando l'interpolazione *Nearest Neighbor* per poi inserire il rumore ed effettuare la compressione.

		quality factor				
		95	55	30	15	1
width in pixels	180					
	150					
	120					
	90					
	70					
	50					
	30					

Figura 3.3 – Immagini di esempio di targhe ceche sintetiche con SNR di 3 dB, nell'asse y la larghezza in pixel, nell'asse x il fattore di qualità JPEG.

Il rumore è principalmente causato dal sensore della fotocamera durante l'acquisizione. Spesso, i modelli di rumore conosciuti sono approssimati dal rumore Gaussiano bianco. Nello studio qui proposto, l'immagine rumorosa è data dalla funzione $f = s + n$ dove s è l'immagine priva di rumore e n è la componente rumorosa. Il rapporto segnale-rumore (SNR) può essere descritto come il rapporto della potenza del segnale e la potenza del rumore. Indichiamo con σ_s^2 la potenza dello spettro del segnale e con σ_n^2 la potenza dello spettro del rumore:

$$SNR = \frac{power(s)}{power(n)} = \frac{\sigma_s^2}{\sigma_n^2}$$

Come anticipato, il training set avrà SNR compreso tra -3 e 20 dB, mentre per il testing set sono usati tre livelli di SNR che corrispondono rispettivamente a livelli di rumore grave, moderato e basso.

L'ultimo step utilizzato per generare le targhe è effettuare la compressione. Viene effettuata una compressione di tipo Lossy e, pertanto, sebbene esistano diversi formati di compressione quello più diffuso è il formato JPEG.

Il fattore di qualità JPEG permette di scambiare la qualità dell'immagine con la dimensione del file.

Nel training set il fattore di qualità JPEG è compreso in un range di 1 e 95 mentre nel testing set sono stati usati cinque livelli di compressione (1, 15, 30, 55 e 95) come è possibile verificare in figura 3.3.

3.3 Architettura CNN

L'architettura della rete neurale presentata da Kaiser et al. [3] adatta la CNN proposta da Lorch et al. [2]. In figura 3.4 viene mostrata l'architettura proposta. Una differenza sostanziale dalla rete sopra analizzata è data dallo strato di input (rosso in figura) in quanto esso è adattato a 44×180 per conformarsi alle

proporzioni delle targhe europee. Per il resto l'architettura rimane invariata alla soluzione precedente.

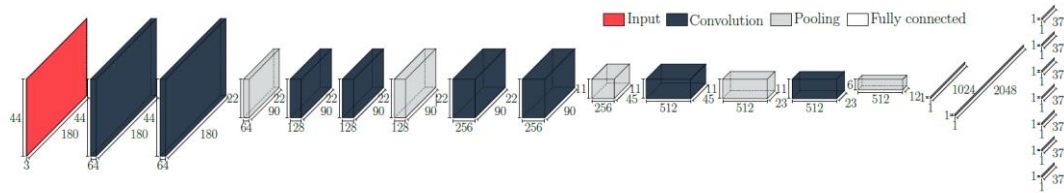


Figura 3.4 – Architettura CNN

Riassumendo si hanno otto livelli convoluzionali (blu in figura) che estraggono le caratteristiche dall'immagine di input con kernel di dimensione 3×3 con passo uno e *zero-padding*.

I livelli di pooling (grigio) con kernel 2×2 sono inseriti dopo il secondo, quarto, sesto, settimo e ottavo livello convoluzionale. I livelli di pooling dispari dimezzano la dimensione spaziale. L'ultimo livello di *pooling* riduce la dimensione spaziale da 11×23 a 6×12 con passo (stride) due e usando *zero-padding* sia a destra che in basso.

Come visto nel paragrafo 2.3, vi sono due livelli *fully-connected* con 1024 e 2048 unità rispettivamente. L'ultimo livello è seguito da sette livelli di output, uno per ogni carattere della targa.

Ogni livello di output è formato da 37 unità per rappresentare 26 lettere dell'alfabeto latino, le cifre da 0 a 9 e il carattere nullo.

Il risultato dei livelli di output è passato alla funzione *softmax* che normalizza il valore di ogni carattere con una distribuzione di probabilità.

Tutti i livelli convoluzionali, compresi i due completamente connessi usano la funzione di attivazione ReLU. I pesi dei livelli convoluzionali e i livelli di output sono inizializzati con il peso di Xavier e i livelli completamente connessi sono inizializzati con una distribuzione normale troncata con media nulla e una variazione standard pari a 0.005. I bias dei livelli di output sono inizializzati a 0 mentre gli altri bias a 0.1.

I parametri vengono aggiornati usando la discesa del gradiente mini-batch con un batch di dimensione di 32. Il tasso di apprendimento inizia a 0.005 e diminuisce esponenzialmente.

L'*overfitting* è ridotto usando i due livelli completamente connessi e i livelli di output con probabilità $p = 0.5$. Infine, la fase di training viene interrotta se l'*accuracy validation* non cambia per 100 000 iterazioni.

3.4 Risultati ottenuti

I risultati verranno analizzati in due contesti: il primo luogo si andrà ad analizzare se la prestazione della rete del training sui dati sintetici sono generalizzabili nel mondo reale quantitativamente ed in secondo luogo, verranno mostrati l'impatto dei tre parametri di distorsione sul tasso di riconoscimento di targhe appartenenti al dataset sintetico; i tre parametri sono: compressione, bassa risoluzione e rumore.

Le previsioni della rete per ogni carattere sono ordinate in base al layer *softmax* di output. Se il carattere corretto rientrerà nelle più probabili verrà considerata una *hit* altrimenti verrà considerato una *miss*. Si considerano quindi i *top-n* come parametro di accuratezza dei singoli caratteri relativi ai tassi di *hit* (tassi di successo).

Nel dettaglio verranno considerate le accuratezze *top-1*, *top-3* e *top-5*.

Per verificare la *performance* della rete su immagini degradate reali è stato effettuato un confronto con lo studio proposto da Spanhel et al. [5] in cui viene utilizzato un dataset nominato "ReId" contenente 76412 immagini reali di targhe europee. Gli autori in particolare hanno posizionato delle videocamere in Full-HD in otto luoghi differenti e in condizioni diverse.

Le immagini sono state ritagliate, convertite a scala di grigi e ridimensionate a 180×44 pixels. Il dataset è formato da immagini leggibili dall'occhio umano ma di bassa qualità.

Dal momento in cui le targhe ceche sono limitate ai sette caratteri ma il dataset contiene anche targhe non-ceche con otto caratteri, è stata aggiunta un'ottava unità di output alla CNN sopra proposta ed è stato effettuato il training con il dataset sintetico. Pertanto, all'ottava posizione verrà restituito un carattere nullo che verrà conteggiato come *hit* nelle targhe a sette caratteri e come *miss* nelle targhe a otto caratteri.

I risultati ottenuti sono riportati nella tabella in figura 3.5.

Gli autori ottengono un'accuratezza di 98.6% sul dataset ReId. Per valutare quanto sia realistico il set di dati sintetico a confronto con i dati del mondo reale, viene effettuato il training sul dataset sintetico e valutato la performance della CNN sul dataset di immagini reali ottenendo una precisione del 89.8% che risulta inferiore del 8% ai risultati ottenuti da Spanhel et al. [5]

La differenza delle prestazioni può essere attribuita a un numero di influenze che non sono considerate dal dataset sintetico. Ad esempio, i dati reali contengono immagini con sfocatura e rotazione prospettica che non sono tenuti in considerazione nel dataset sintetico.

Per risolvere questo gap è stato effettuato fine-tune sulle immagini del training set con un tasso di apprendimento del 0.005. Con un fine-tuning top-1 si ottiene una accuratezza di 97.3% che sebbene sia inferiore ai risultati ottenuti da Spanhel et al. [5], è sufficientemente alto per l'analisi di immagini degradate reali.

Method	top-1	top-3	top-5
Spanhel <i>et al.</i> [22]	98.6%	-	-
CNN synthetic	89.8%	95.6%	97.3%
CNN fine-tuned	97.3%	98.7%	99.0%




Figura 3.5 – Accuratezza top-1, top-3, top-5 (sx) sul dataset di targhe ceche (dx)

Nella seconda parte della valutazione della *performance* della CNN verrà effettuata l'analisi sull'impatto che subisce l'accuratezza al variare dei parametri di compressione, SNR e risoluzione dell'immagine.

Le valutazioni sono effettuate sul set di dati sintetici. Lo scopo è quello di valutare l'influenza della degradazione dell'immagine sul tasso riconoscimento dei caratteri al variare dei parametri sopracitati.

Verrà valutato, inoltre la similarità tra i vari caratteri e la posizione degli stessi. La media di riconoscimento sul dataset sintetico su tutte le targhe, tutte le posizioni e tutti i livelli di degradazione si ottiene *top-1*: 85.4%, *top-3*: 93.1%, *top-5*: 96.0%.

Per quanto riguarda l'incidenza della compressione JPEG sul riconoscimento dei caratteri viene mostrata in figura 3.6 l'influenza del tasso di compressione sul tasso di riconoscimento della targa. Si evincono nel grafico l'accuratezza *top-1* di diversi livelli di compressione in relazione alla larghezza della targa in pixel. I risultati fanno riferimento alla media su tutti i livelli di rumore.

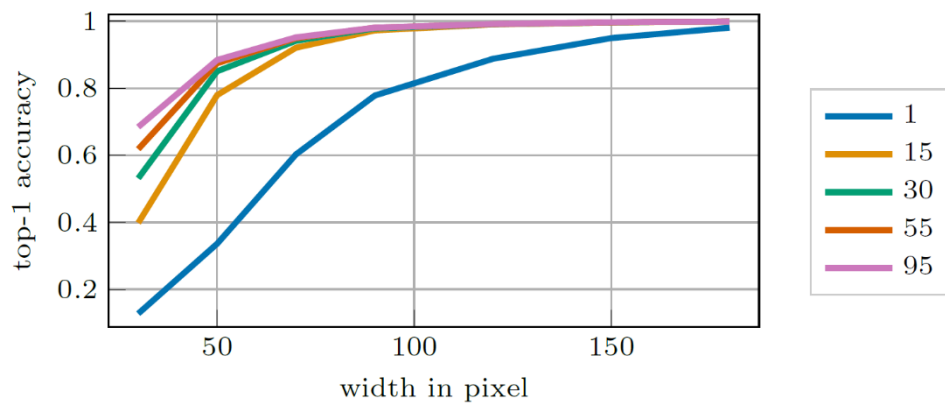


Figura 3.6 – Accuratezza *top-1* in relazione alla larghezza dell'immagine in pixel.

È possibile verificare dal grafico che la precisione di riconoscimento ha un impatto considerevole su immagini con larghezza inferiori a 70 pixel. Possiamo affermare che una forte compressione rimuove informazioni particolarmente importanti per la distinzione dei caratteri con immagini a risoluzione molto bassa,

pertanto è necessario effettuare un compromesso tra risoluzione dell'immagine e compressione JPEG.

La performance aumenta con una qualità JPEG tra 55 e 95 ma i caratteri sono appena distinguibili, è necessario avere una qualità di compressione maggiore di 90 pixels per ottenere un'accuratezza accettabile.

In figura 3.7 viene analizzato l'impatto che subisce l'accuratezza della rete con l'aumentare del rumore nell'immagine, nel dettaglio con livelli di rumore pari a -3 dB, 3 dB, 20 dB. Ogni grafico è suddiviso in base alla larghezza dell'immagine della targa e al fattore di qualità JPEG.

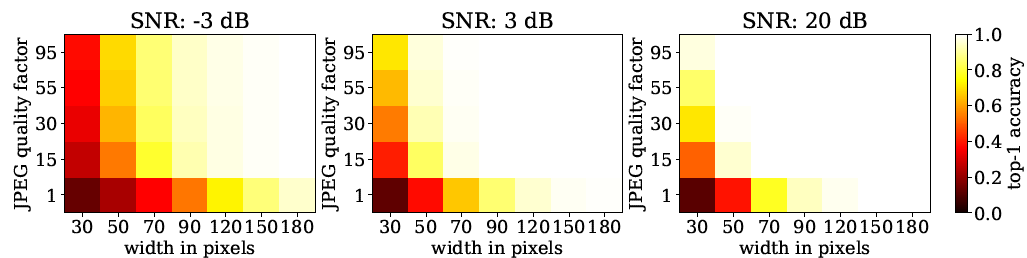


Figura 3.7 – Accuratezza *top-1* in relazione a diversi livelli di SNR nell'immagine

Nella tabella di seguito vengono riportati i valori esatti in relazione a SNR, fattore di qualità e larghezza dell'immagine in pixels.

SNR [dB]	quality factor	width in pixels						
		30	50	70	90	120	150	180
-3	95	0.38	0.69	0.86	0.94	0.98	0.99	1.00
	55	0.36	0.67	0.86	0.94	0.97	0.99	1.00
	30	0.33	0.63	0.84	0.94	0.97	0.99	1.00
	15	0.28	0.55	0.79	0.92	0.97	0.99	1.00
	1	0.14	0.23	0.37	0.54	0.73	0.87	0.95
3	95	0.71	0.96	0.99	1.00	1.00	1.00	1.00
	55	0.64	0.95	0.99	1.00	1.00	1.00	1.00
	30	0.55	0.93	0.99	1.00	1.00	1.00	1.00
	15	0.41	0.84	0.97	1.00	1.00	1.00	1.00
	1	0.13	0.37	0.66	0.86	0.95	0.98	1.00
20	95	0.96	1.00	1.00	1.00	1.00	1.00	1.00
	55	0.85	1.00	1.00	1.00	1.00	1.00	1.00
	30	0.71	0.99	1.00	1.00	1.00	1.00	1.00
	15	0.51	0.95	1.00	1.00	1.00	1.00	1.00
	1	0.12	0.39	0.78	0.93	0.98	1.00	1.00

Figura 3.8 – Accuratezza *top-1* in relazione a diversi livelli di SNR nell'immagine, fattore di qualità e larghezza in pixels.

I risultati mostrano che la larghezza delle immagini delle targhe deve essere superiore a 30 pixel per ottenere risultati affidabili.

Anche la posizione dei caratteri incide sull'accuratezza della rete. In figura 3.9 viene mostrata l'accuratezza top-1 nelle sette posizioni. Nel primo grafico viene messa in relazione la posizione della targa valutata con diversi tassi di compressione e nel secondo grafico in basso viene messa in relazione la larghezza in pixel delle targhe e la posizione dei caratteri. In entrambi i grafici la posizione tre risulta particolarmente difficoltosa da riconoscere in quanto tale posizione può contenere tutte le cifre e 22 lettere nelle targhe ceche. La migliore accuratezza si ottiene alla posizione due in quanto può contenere meno caratteri possibili come già analizzato. Ugualmente per la posizione sei e sette che possono contenere solo cifre.

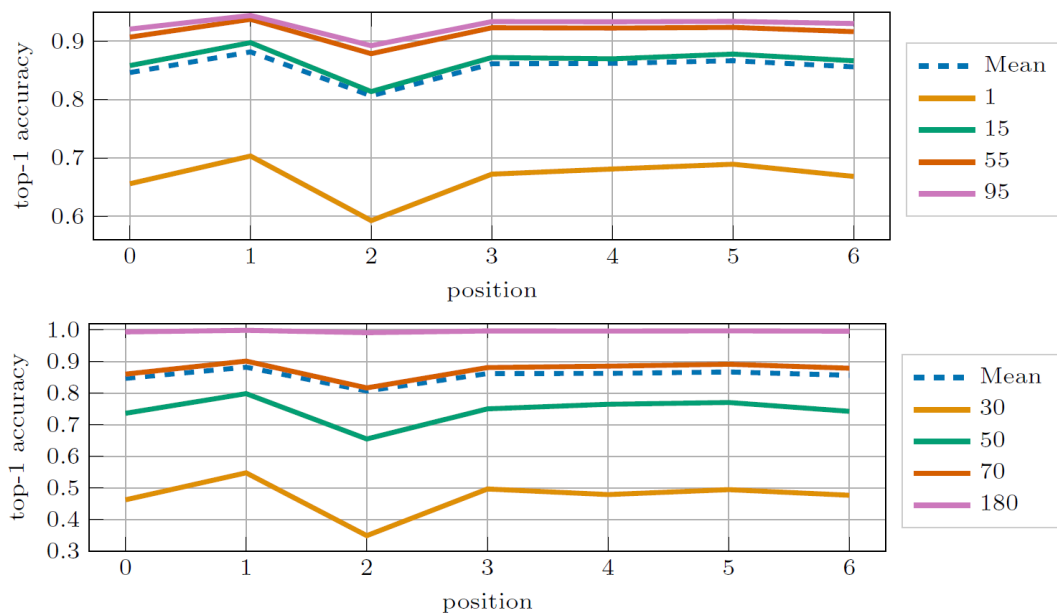


Figura 3.9 - In alto accuratezza top-1 analizzando posizione dei caratteri e diversi tassi di compressione, in basso accuratezza top-1 analizzando posizione dei caratteri e larghezza in pixel.

In aggiunta, alcuni caratteri al di là della loro posizione risultano maggiormente ostici per il riconoscimento, come si denota in figura 3.10 in cui viene mostrata l'accuratezza massima (top-1) per ogni carattere facendo una media dei risultati ottenuti su tutte le posizioni.

In p1, p2, p3 viene mostrata la previsione con la corrispondenza dei vari caratteri e le confidenze medie per ogni carattere rispettivamente in c1, c2, c3.

char	top-1	p1	p2	p3	c1	c2	c3	char	top-1	p1	p2	p3	c1	c2	c3
0	0.86	0	8	6	0.82	0.03	0.03	H	0.86	H	M	U	0.82	0.03	0.03
1	0.89	1	3	7	0.87	0.03	0.02	I	0.83	I	1	T	0.79	0.03	0.03
2	0.88	2	7	3	0.86	0.03	0.03	J	0.91	J	U	Z	0.88	0.02	0.01
3	0.86	3	2	1	0.83	0.03	0.03	K	0.83	K	E	A	0.81	0.03	0.02
4	0.91	4	6	2	0.89	0.02	0.01	L	0.90	L	E	C	0.87	0.03	0.02
5	0.84	5	6	8	0.81	0.04	0.03	M	0.89	M	H	B	0.86	0.04	0.01
6	0.79	6	8	5	0.76	0.08	0.05	N	0.82	N	H	8	0.78	0.03	0.03
7	0.90	7	2	1	0.87	0.04	0.02	P	0.87	P	E	M	0.85	0.02	0.01
8	0.82	8	6	9	0.78	0.06	0.03	R	0.75	R	8	6	0.73	0.03	0.02
9	0.82	9	8	0	0.80	0.05	0.03	S	0.80	S	B	C	0.78	0.02	0.02
A	0.86	A	K	B	0.84	0.01	0.01	T	0.87	T	Z	Y	0.84	0.02	0.02
B	0.79	B	8	H	0.76	0.03	0.06	U	0.83	U	H	J	0.81	0.04	0.02
C	0.84	C	E	L	0.81	0.03	0.02	V	0.83	V	9	8	0.80	0.02	0.02
D	0.72	D	0	U	0.68	0.09	0.03	X	0.78	X	Y	K	0.75	0.02	0.02
E	0.85	E	C	L	0.81	0.03	0.02	Y	0.79	Y	T	1	0.76	0.04	0.03
F	0.81	F	P	E	0.78	0.05	0.04	Z	0.85	Z	2	T	0.83	0.03	0.02

Figura 3.10 – Accuratezza top-1 per ogni carattere.

La similarità fra i vari caratteri viene misurata dalle feature estratte dai livelli convoluzionali e possono essere la direzione, la posizioni dei tratti e le proiezioni orizzontali. Ad esempio, i caratteri B e 8 risultano difficili da distinguere, o come è possibile vedere in figura 3.11, l'H, W, U hanno proiezioni orizzontali simili e risultano difficilmente distinguibili.

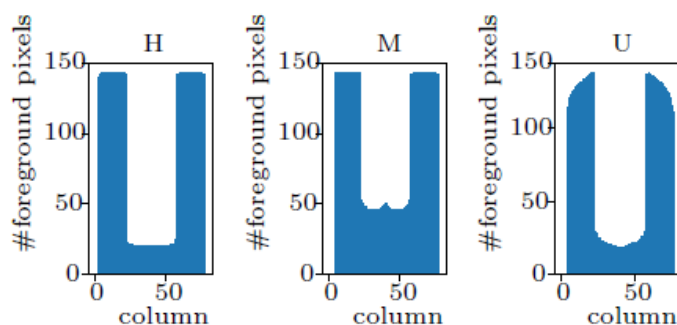


Figura 3.11 – Similarità delle proiezioni orizzontali dei caratteri H, M e U.

La compressione influisce anche sulle similarità dei caratteri come è possibile verificare in figura 3.12 sui caratteri C e P. Notiamo che la lettera C viene confusa con la lettera E per tutti i fattori di qualità. Solo la previsione tre cambia con un fattore di qualità superiore. D'altro canto, la lettera P è confusa con le lettere E, M ed F per i fattori di qualità 1, 15, e 95 rispettivamente.

quality factor	char	p1	p2	p3	c1	c2	c3	char	p1	p2	p3	c1	c2	c3
1	C	C	E	U	0.58	0.04	0.04	P	P	E	H	0.62	0.04	0.03
15	C	C	E	Z	0.83	0.03	0.02	P	P	M	F	0.87	0.02	0.01
95	C	C	E	L	0.90	0.02	0.01	P	P	F	E	0.92	0.01	0.01

Figura 3.12 – Similarità dei caratteri C e P in relazione a vari livelli di compressione

3.5 Conclusioni

Dopo aver effettuato l'analisi sul riconoscimento delle targhe sintetiche ceche in cui sono presenti immagini soggette a compressione JPEG e forte degradazione è possibile affermare che per immagini a bassa risoluzione la compressione ha un impatto maggiore rispetto a immagini con larghezza in pixel maggiori. Un fattore di qualità uno porta ad una drastica diminuzione del tasso di riconoscimento.

Targhe ceche sintetiche possono essere ricostruite in modo affidabile se la larghezza delle immagini è superiore a 30 pixel, l'SNR è superiore -3 dB e il fattore di qualità JPEG è almeno 15.

Infine, la risoluzione ha un impatto maggiore rispetto alla compressione quando il carattere ha più probabilità di scelta, come alla terza posizione delle targhe ceche.

Capitolo 4

Rete Neurale di Denoising e di Lettura di targhe degradate^[4]

Il lavoro svolto da Gianmaria Rossi, Marco Fontani e Simone Milani^[4] pone come obiettivo, così come i lavori visti nei capitoli precedenti, l'interpretazione ed il riconoscimento delle targhe fortemente degradate e la riduzione del rumore.

Nel dettaglio vengono accoppiate due reti neurali convoluzionali dove la prima rete produce una versione *denoised* dell'immagine originale e la seconda rete effettua una stima dei caratteri all'interno della targa generando un vettore di probabilità.

Rispetto alle soluzioni viste precedentemente, il sistema fornisce sia una targa *denoised* che una previsione dei caratteri.

4.1 Caratteristiche principali

Lo studio svolto mira ad aumentare l'applicabilità del lavoro di Lorch et al. [2] con l'introduzione prospettica nelle immagini del training set e migliorare le prestazioni aggiungendo una rete di riduzione del rumore della targa in input.

Vengono, dunque, utilizzate due reti CNN: la prima per effettuare il *denoising* dell'immagine mentre la seconda effettua il riconoscimento ed una stima dei caratteri più probabili della targa analizzata.

L'approccio è stato testato e addestrato solo su targhe italiane.

Dal momento in cui quest'ultima affermazione rappresenta una limitazione nell'utilizzo del sistema proposto, è stato definito un metodo che consente di generare targhe di differenti paesi senza il bisogno di raccogliere dati reali per il training. Pertanto, è stato sviluppato un algoritmo di generazione di dataset il quale in un primo momento crea targhe sintetiche e realistiche e successivamente applica ad esse diverse degradazioni aumentando la diversità del set di dati di addestramento e, difatti, la robustezza finale del sistema.

Il sistema proposto funziona in due fasi: la prima utilizza un *auto-encoder* CNN per generare una versione *denoised* dell'immagine dopo di ciò sia l'immagine originale che la versione *denoised* vengono mandati alla seconda CNN che esegue una interpretazione dei caratteri e ne effettua il riconoscimento.

4.2 Dataset

Per effettuare il *training* della rete è stato sviluppato un dataset formato da 20 mila targhe sintetiche utilizzando stringhe casuali di caratteri seguendo il layout delle targhe italiane.

Le immagini hanno dimensioni di 160×40 pixel che non è la dimensione esatta delle targhe italiane ma rappresenta un buon compromesso con le proporzioni utilizzate negli altri stati europei.

Mostriamo di seguito, in figura 4.1, degli esempi di targhe sintetiche realizzate.

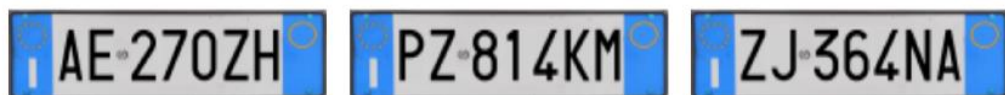


Figura 4.1 - Esempio di targa sintetica italiana

Al fine di aumentare la variabilità di targhe sintetiche sono state applicate diverse trasformazioni alle immagini, tra cui:

- cambiamenti casuali di illuminazione: dopo aver convertito l'immagine nello spazio colore HSV, quindi per i pixel di sfondo è stato aggiunto un valore causale compreso tra -20 e 20 al primo canale e un valore compreso tra -10 e 10 al terzo canale;
- ombre casuali: viene aggiunta l'ombra all'immagine selezionando una regione della targa e rendendola più scura. Vengono selezionati gli angoli in alto a dx o sx e poi in modo casuale vengono scelti gli altri due punti per creare la regione;
- ritaglio casuale: viene ritagliata in modo casuale l'immagine mantenendo ben visibili i caratteri. Questo fa sì che la rete non impari che un carattere è sempre nella stessa posizione.

Viene, inoltre, creata una maschera binaria della targa in modo tale che i pixel che compongono i caratteri viene assegnato valore 0 mentre ai pixel dello sfondo valore 1 (Figura 4.3). Questa maschera verrà utilizzata nella funzione di perdita della rete di *denoising* che verranno esaminate in un secondo momento.

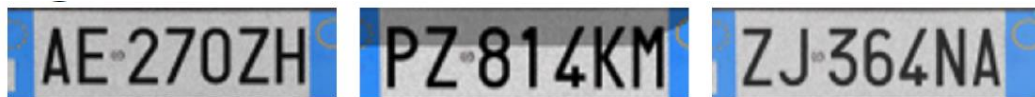


Figura 4.2 – Trasformazioni sulle immagini

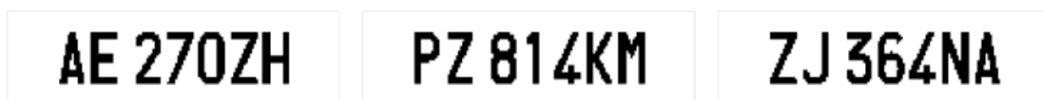


Figura 4.3 – Maschera binaria ottenuta

Successivamente per creare un effetto quanto più verosimile a situazioni reali ed imitare la distorsione introdotta dai sistemi di videosorveglianza, vengono applicate una serie di trasformazioni alle targhe sintetiche.

Le operazioni svolte in ordine sono:

1. la targa può o meno essere ridimensionata ad una risoluzione inferiore. A tal fine sono state utilizzate tre diverse risoluzioni inferiori: 33×8 , 60×15 , $120 \times 30 \text{ pixel}$. Il metodo di interpolazione utilizzato per il ridimensionamento è INTER_AREA mentre per l'aumento dell'immagine viene utilizzato INTER_CUBIC;
2. è stata introdotta la distorsione prospettica tramite la funzione *warpPerspective* di OpenCV. L'immagine è stata ridimensionata alla sua dimensione originale ed è stata applicata la deformazione inversa;
3. è stato applicato un kernel di sfocatura gaussiana utilizzando la funzione *GaussianBlur* di OpenCV con tre differenti dimensioni del kernel ovvero 1, 3 o 5. La deviazione standard Gaussiana è calcolata nel seguente modo:

$$\sigma = 0.3 \cdot ((ksize - 1)) \cdot 0.5 - 1) + 0.8$$

4. infine, viene aggiunta alla targa un rumore gaussiano casuale di quattro diverse intensità a seconda del valore della media μ e della varianza σ . Verranno utilizzati come valori no-noise: $\mu = \sigma = 1$, low-noise: $\mu = \sigma = 70$, mid-noise: $\mu = \sigma = 500$, high-noise: $\mu = \sigma = 2000$.



Figura 4.4 – Targa sintetica ottenuta con varie degradazioni artificiali

Sono state ottenute in totale 192 differenti combinazioni di trasformazioni applicate al dataset originale formato da 20 mila targhe ottenendo, difatti 3 840 000 targhe.

Poiché le targhe sono sintetiche, per testare la precisione del modello è stato creato anche un piccolo dataset di 884 targhe reali. A tal fine è stato effettuato il *download* delle targhe disponibili online al sito *platesmania.com* ed in seguito, tramite il software License Plate Extractor (LPEX), l'immagine è stata ritagliata.

Le targhe non sono state ritagliate al limite, per simulare uno scenario reale in cui l'utente non ha prestato molta attenzione alla creazione dell'input per le reti neurali.

Successivamente, dopo aver estratto la maschera binaria, sono state effettuate tutte le operazioni sopra elencate per il dataset sintetico ottenendo infine un dataset di prova di 169728 immagini.

La targa viene quindi, convertita in scala di grigi per soddisfare i requisiti della rete che andremo ad analizzare di seguito.

Il dataset viene suddiviso in due parti: l' 80% del dataset viene utilizzato per l'addestramento e la restante parte del dataset viene utilizzata per la validazione dei parametri.

4.3 Architettura rete

L'approccio sviluppato da Rossi et al. [4] si avvale di due CNN: una rete di *denoising* che produce un'immagine meno rumorosa della targa in input e la rete di lettura che genera un vettore di probabilità per ogni carattere della targa.

Analogamente al lavoro svolto da Lorch et al. [2] il sistema sviluppato non richiede la segmentazione dei caratteri.

Per quanto riguarda la rete neurale di *denoising* si basa su un'architettura U-Net, ovvero un auto-encoder in cui ogni blocco del percorso di encoder è concatenato con l'input del blocco corrispondente del percorso del decoder.

La rete possiede quattro blocchi per l'encoder e tre per il decoder.

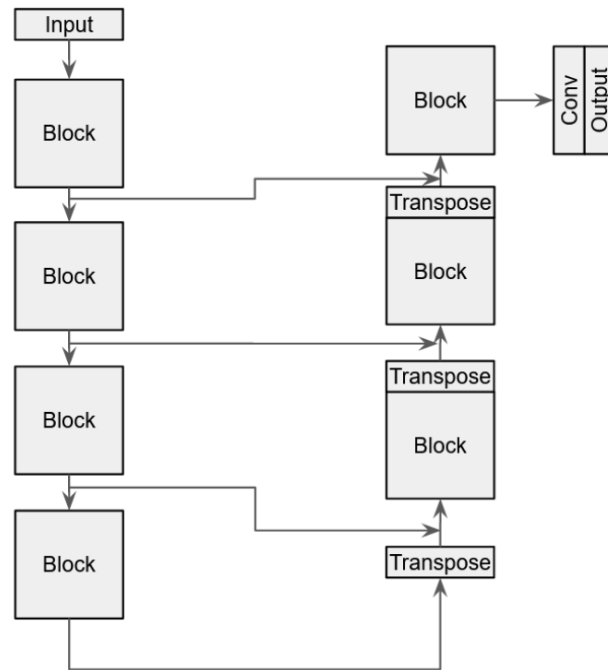


Figura 4.4 – Architettura rete di Denoising

Ogni blocco del percorso di encoder e decoder viene realizzato concatenando due volte i seguenti livelli: uno strato convoluzionale con filtri n_f con dimensione del kernel di 3×3 e passo 1; un livello di normalizzazione batch ed un livello di attivazione che utilizza la funzione di attivazione ReLU. Il valore di n_f è uguale a 48 nel primo blocco, 96 nel secondo, 144 nel terzo e 192 nell'ultimo percorso dell'encoder mentre 144, 96 e 48 nel percorso del decoder. Dopo ogni blocco, ad eccezione dell'ultimo, viene applicato un livello di max-pooling che decrementa la dimensione della mappa di *feature*. Partendo da un input di dimensione di 160×40 , dopo ogni blocco avremo una *feature map* di 80×20 , 40×10 , 20×5 . Dopo ogni livello di pooling è presente anche un livello di dropout con probabilità $p = 0.05$ per ignorare un nodo.

Nel percorso del decoder è presente anche uno strato convoluzionale trasposto per raddoppiare la dimensione della mappa delle feature.

Pertanto, viene concatenato il risultato con l'output del blocco corrispondente del percorso dell'encoder. La concatenazione viene data come input ad un blocco con

la stessa architettura utilizzata nel percorso dell'encoder e viene utilizzato un livello di Dropout dopo ogni blocco tranne per l'ultimo. Difatti, l'ultimo è uno strato convoluzionale con un singolo kernel 1×1 e passo 1 con funzione di attivazione sigmoide. Questo darà un output della stessa dimensione dell'input ma con un singolo canale.

Per quanto riguarda la rete di lettura essa opera solo su immagini a scala di grigi ed elabora una concatenazione tra la targa originale e la targa denoised (la targa di output della rete di Denoising). La rete di lettura è formata da otto strati convoluzionale con dimensione del kernel di 3×3 ed un numero di filtri pari a: 64, 64, 128, 128, 256, 256, 512, 512. Ognuno è seguito da una funzione di attivazione ReLU ed è presente un livello max-pooling dopo la seconda, quarta, sesta, settima e ottava funzione di attivazione. I livelli di max-pooling in posizione dispari decrementano di due la dimensione della *feature map*. Dopo l'ultimo livelli di pooling ci sono due livelli completamente connessi, il primo con 1024 nodi e il secondo con 2048 nodi. C'è anche un livello di esclusione dopo ogni livello completamente connesso ciascuno con una probabilità di ignorare un nodo uguale a $p = 0.5$

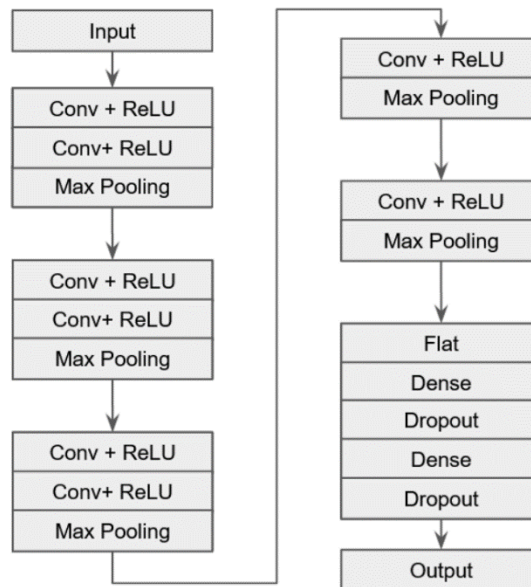


Figura 4.5 – Architettura rete di lettura

Conformemente a quanto effettuato da Lorch et al. [2], i pesi dei livelli convoluzionali e il sette strati densi finali sono inizializzato con il peso di Xavier. Per i primi due livelli densi viene utilizzata una distribuzione normale troncata con media zero e deviazione standard pari a 0.005. Il bias iniziale è impostato a 0.1 per tutti i livelli ad eccezione dei primi due strati densi che è uguale a 0. L'output è costituito da sette strati densi, ognuno corrispondente a uno dei sette caratteri di una targa italiana. Ciascuno dei livelli di output ha 37 nodi: 26 per l'alfabeto latino, 10 per i numeri e 1 per lo spazio usato nei casi di targhe con numero di carattere diversi. Dal momento in cui le moderne targhe italiane sono sempre formate da 7 caratteri la probabilità di utilizzare un carattere vuoto è quasi 0.

In figura 4.6 viene mostrata la probabilità di ogni singolo carattere di una targa italiana.

prediction 1	prediction 2	prediction 3	prediction 4	prediction 5
A 0.9927	X 0.0054	L 0.0013	J 0.0002	V 0.0001
X 1.0000	A 0.0000	K 0.0000	Y 0.0000	N 0.0000
6 0.9985	5 0.0014	8 0.0001	3 0.0000	0 0.0000
7 1.0000	2 0.0000	1 0.0000	0 0.0000	9 0.0000
2 0.9812	7 0.0186	0 0.0001	1 0.0000	3 0.0000
C 0.9969	G 0.0031	Z 0.0000	D 0.0000	J 0.0000
X 0.9999	A 0.0000	K 0.0000	Y 0.0000	V 0.0000

Figura 4.6 – Previsione dei caratteri di una targa italiana posta in alto in figura

4.4 Risultati ottenuti

Il training del sistema è diviso in due parti: la prima parte viene effettuato il training della sola rete di *denoising* e nella seconda viene addestrata la rete di lettura sia con le targhe degradate sia con la versione *denoising* ottenuta dalla prima rete (che è stata precedentemente addestrata) concatenando in un unico input.

Difatti così facendo, le informazioni presenti in entrambe le immagini vengono combinate.

A tal fine è stato utilizzando il framework TensorFlow in combinazione con le API di Keras in un pc con due RTX 2080 Ti che lavorano in parallelo.

Per il training della rete di *denoising* è necessario avere l'immagine originale, la versione degradata ottenuta tramite le varie operazioni sopracitate e la maschera ottenuta durante la creazione del dataset sintetico.

Il lavoro è stato svolto con batch di 256 campioni che in un ambiente multi- GPU è suddiviso in modo uniforme su tutte le GPU disponibili in modo tale che ogni GPU operi con un batch di 128 campioni.

L'ottimizzatore utilizzato è Adam con un learning rate iniziale pari a 0.05 ed il training è stato interrotto dopo 1200 iterazioni.

La funzione di perdita utilizzata per la retropropagazione dell'errore – *backpropagation* è la combinazione di tre differenti perdite:

$$\mathcal{L} = \lambda_1 * \mathcal{L}_{mse} + \lambda_2 * \mathcal{L}_{wass} + \lambda_1 * \mathcal{L}_{mask}$$

Le parziali perdite vengono calcolate:

- \mathcal{L}_{mse} è l'errore quadratico medio (MSE) calcolato tra l'output della rete di Denoising e la targa originale.
- $\mathcal{L}_{wass} = 1 + \frac{\sum_{i,j} (y_{i,j}^{true} * y_{i,j}^{pred})}{N}$ dove N è il numero di pixel (160×40), $y_{i,j}^{true}$ è il valore del pixel in posizione i, j della matrice corrispondente alla targa originale e $y_{i,j}^{pred}$ è il valore del pixel in posizione i, j della previsione della rete di Denoising.
- \mathcal{L}_{mask} è calcolato come $\mathcal{L}_{mask} = 1 + \rho - b$ dove $\rho = \frac{\sum_{i,j} (y_{i,j}^{true} * \overline{y_{i,j}^{mask}})}{\sum_{i,j} y_{i,j}^{mask}}$ e $y_{i,j}^{mask}$ è il pixel in posizione i, j della maschera dell'immagine e

$$b = \frac{\sum_{i,j} (y_{i,j}^{pred} * y_{i,j}^{mask})}{\sum_{i,j} y_{i,j}^{mask}}$$

- I valori di λ_1, λ_2 e λ_3 sono rispettivamente 30, 0.2, 0.2. il motivo per cui λ_1 ha valore maggiore è perchè \mathcal{L}_{mse} è minore rispetto alle altre due perdite infatti $\mathcal{L}_{mask}, \mathcal{L}_{wass} \approx 1$ e $\mathcal{L}_{mse} \approx 0.035$.

Per verificare la qualità dell'output della rete di *denoising* è stato utilizzato *Complex Wavelet Structural Similarity Index Measure (CW-SSIM)*[6].

Per il training della rete di lettura l'ottimizzatore utilizzato è lo Standard Descent (SGD) con un learning rate iniziale pari a 0.05, il batch globale è di 64 campioni e ogni GPU lavora con batch di 32 campioni.

La funzione di perdita è calcolata sul *validation set* ogni 1000 iterazioni sul training set ed il training viene fermato dopo che la *validation loss* non è migliorata per 100 volte.

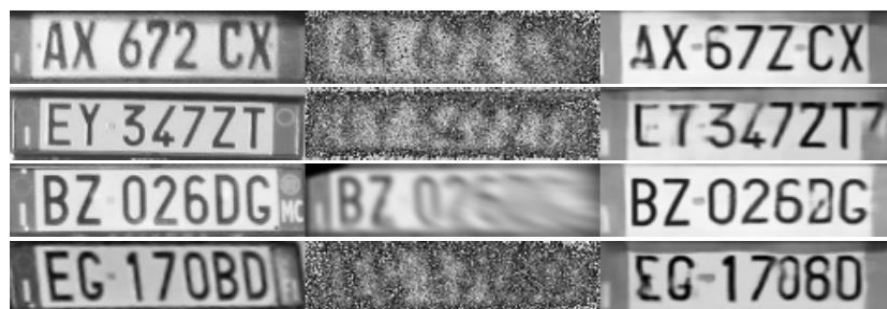
Per calcolare la perdita è stato creato 7 *one-hot encoding* di 37 elementi, uno per ogni carattere della targa e successivamente sommando la perdita di entropia incrociata tra l'output della rete e l'etichetta vera. $\mathcal{L} = l_1 + l_2 + l_3 + l_4 + l_5 + l_6 + l_7$ con $l_n = \sum_{i=1}^{37} (y_i * \log(\hat{y}_i))$ per $n = 1, \dots, 7$ dove \hat{y}_i è l' i -esimo valore dell'output e y_i è l' i -esimo valore dell'etichetta.

Per verificare l'accuratezza dell'output vengono utilizzate tre etichette: Top-1, Top-3, Top-5.

La precisione Top-n su un insieme S di campioni ha sette valori, uno per ciascun carattere della targa; in ogni posizione sommiamo il numero di occorrenze che il carattere corretto, dato dall'etichetta, si trova nei primi n valori dell'output e successivamente verrà effettuata la media (Figura 4.7).

	position 1	position 2	position 3	position 4	position 5	position 6	position 7	Average
TOP 1	0.93005	0.94324	0.96183	0.95992	0.95232	0.91980	0.86653	0.93338
TOP 3	0.97298	0.98528	0.99051	0.99095	0.98781	0.96999	0.94586	0.97762
TOP 5	0.98438	0.99160	0.99487	0.99560	0.99499	0.98020	0.96488	0.98664

Figura 4.7 – Accuratezza media su test set



prediction 1	prediction 2	prediction 3	prediction 4	prediction 5
A 0.9927	X 0.0054	L 0.0013	J 0.0002	V 0.0001
X 1.0000	A 0.0000	K 0.0000	Y 0.0000	N 0.0000
6 0.9985	5 0.0014	8 0.0001	3 0.0000	0 0.0000
7 1.0000	2 0.0000	1 0.0000	0 0.0000	9 0.0000
2 0.9812	7 0.0186	0 0.0001	1 0.0000	3 0.0000
C 0.9969	G 0.0031	Z 0.0000	D 0.0000	J 0.0000
X 0.9999	A 0.0000	K 0.0000	Y 0.0000	V 0.0000
prediction 1	prediction 2	prediction 3	prediction 4	prediction 5
E 0.8352	L 0.1498	C 0.0083	F 0.0031	K 0.0005
Y 0.9994	T 0.0002	X 0.0001	L 0.0001	V 0.0001
3 0.9963	5 0.0035	1 0.0001	0 0.0001	9 0.0000
4 0.9999	7 0.0000	8 0.0000	2 0.0000	1 0.0000
7 0.9997	2 0.0003	0 0.0000	1 0.0000	4 0.0000
Z 0.9977	T 0.0011	M 0.0004	P 0.0003	Y 0.0001
T 0.9999	Z 0.0000	Y 0.0000	X 0.0000	G 0.0000
prediction 1	prediction 2	prediction 3	prediction 4	prediction 5
B 1.0000	R 0.0000	D 0.0000	P 0.0000	S 0.0000
Z 1.0000	P 0.0000	C 0.0000	X 0.0000	L 0.0000
0 1.0000	8 0.0000	6 0.0000	2 0.0000	9 0.0000
2 1.0000	5 0.0000	8 0.0000	9 0.0000	4 0.0000
6 0.9999	5 0.0000	8 0.0000	0 0.0000	3 0.0000
D 0.9185	B 0.0814	Z 0.0000	S 0.0000	E 0.0000
G 0.9999	C 0.0000	S 0.0000	W 0.0000	Z 0.0000
prediction 1	prediction 2	prediction 3	prediction 4	prediction 5
E 0.9877	S 0.0113	X 0.0003	L 0.0001	R 0.0001
G 0.9994	C 0.0005	D 0.0000	W 0.0000	F 0.0000
1 1.0000	3 0.0000	5 0.0000	0 0.0000	9 0.0000
7 0.9999	1 0.0000	2 0.0000	9 0.0000	0 0.0000
0 0.9778	8 0.0099	6 0.0090	3 0.0015	5 0.0006
B 0.9995	D 0.0004	R 0.0001	P 0.0000	S 0.0000
D 0.9999	B 0.0001	R 0.0000	J 0.0000	W 0.0000

Figura 4.8 – Previsioni dei caratteri delle targhe in alto in figura

4.5 Conclusioni

La soluzione proposta, rispetto alle soluzioni viste precedentemente, consente di avere una versione *denoised* della targa per una visualizzazione della stessa tramite la stessa è possibile effettuare un confronto con il risultato ottenuto con la rete di lettura.

Infatti, spesso l'utilizzo di metodologie basate sul *deep learning* per applicazioni forensi risulta problematico perché si basano su dati di informazioni esterni al caso giudiziario specifico e la loro interpretazione risulta essere ancora limitata.

Pertanto, per tentare di risolvere parzialmente questa problematica la rete *denoised* consente di ottenere e fornire all'analista una targa intermedia nitida che permette di ricontrollare il risultato ottenuto aumentandone l'affidabilità con valenza probatoria.

I risultati ottenuti dimostrano che anche nei casi in cui la targa è molto degradata e l'occhio umano non è in grado di riconoscere con esattezza i caratteri, attraverso l'uso delle reti neurali è possibile ottenere un'immagine dove è possibile poter estrarre dettagli e anche rilevare ogni carattere con una buona attendibilità.

Si è ottenuta infatti una accuratezza media della classificazione dei caratteri del 93%.

Il lavoro proposto ha inoltre mostrato come poter generare *dataset* da utilizzare per il training tramite un algoritmo di generazione di *dataset*.

Quest'ultimo consente di creare targhe sintetiche e realistiche e applica ad esse diverse degradazioni aumentando la diversità del set di dati di addestramento e, difatti, la robustezza finale del sistema.

Capitolo 5

Sperimentazioni su dati di targhe italiane

I lavori precedentemente analizzati nei capitoli 2 e 3 si basano su algoritmi di *deep learning* su *dataset* di targhe fortemente degradate rispettivamente statunitensi e ceche.

Pertanto, con l'obiettivo di introdurre un nuovo studio per il riconoscimento di targhe, si è ritenuto opportuno effettuare varie sperimentazioni su *dataset* di targhe italiane.

In questo capitolo sono esposti diversi esperimenti effettuati basandosi sulla rete proposta da Lorch et. al ^[2] ed il set di dati prodotto da Rossi et al.^[4]

Lo scopo principale delle sperimentazioni effettuate consiste nell'aumentare l'applicabilità della rete di Lorch a contesti reali in cui le immagini acquisite risultano di difficoltosa leggibilità e identificazione.

5.1 Dati e impostazioni sperimentali

Per le varie sperimentazioni è stato utilizzato Python 3.7 in combinazione con il framework TensorFlow v. 2.1 su un pc con Ubuntu 18.04.4 con quattro GPU GK210GL.

Il primo approccio che è stato effettuato consiste nell'aggiungere all'immagine delle targhe in figura 4.2 una notevole distorsione prospettica al fine di simulare contesti reali in cui l'immagine non viene acquisita frontalmente.

Si ritiene opportuno far notare che, come descritto nel paragrafo 4.2, alle immagini sono state già applicate diverse trasformazioni come cambiamenti casuali di illuminazione, ombre e ritagli casuali.

Pertanto, utilizzando le 22.000 immagini del dataset originale ridimensionate da 160×40 a 100×50 pixel, è stata effettuata *Data Augmentation* applicando 10 tipologie di distorsione prospettica differenti ottenendo in totale 220.000 immagini.



Figura 5.1 – Distorsioni Prospettiche

In un secondo momento, per simulare casi reali in cui l'immagine acquisita presenti del rumore impulsivo sono stati aggiunti tre gradi diversi di rumore ottenendo 660.000 immagini. A tal fine, è stata utilizzata la libreria *imgaug* e nel dettaglio la funzione *CoarseDropout* con una frequenza di rumore del 10%, 30%, e 50% dei pixel dell'immagine.

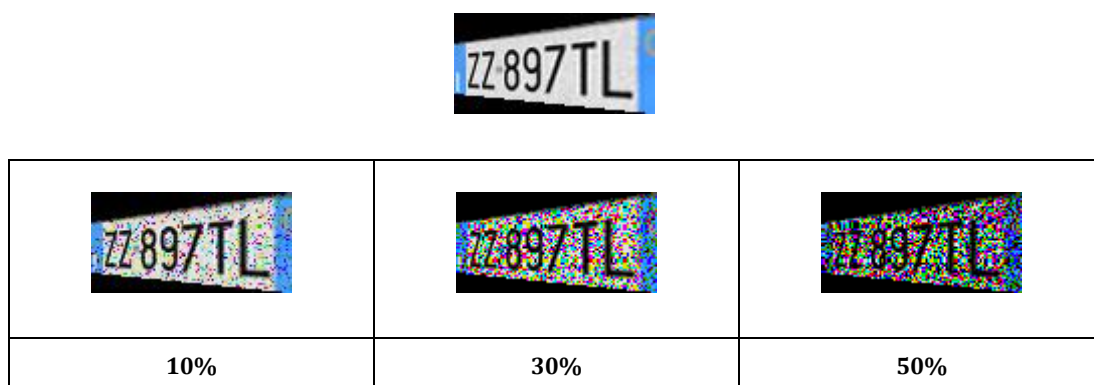


Figura 5.2 – Rumore impulsivo

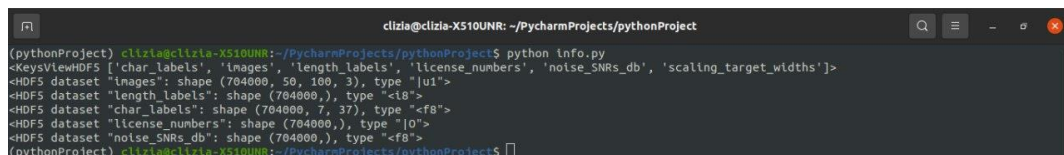
Dunque, il nuovo dataset è formato da 880.000 immagini di cui l'80% sono state utilizzate per il training della rete, il 10% per la validazione e 10% per testare la rete.

Le immagini sono state successivamente trasformate in HDF5 - *Hierarchical Data Format* che consente di ottimizzare l'archiviazione dei *dataset* di *training*, *validation* e *testing*.

Le immagini e le etichette sono state archiviate in un singolo file HDF5 ed al suo interno le immagini vengono memorizzate in un dataset denominato "*images*" e le etichette vengono archiviate in un dataset denominato "*char_labels*".

Pertanto, il dataset delle immagini avrà dimensione [num_images, 50, 100, 3] mentre il dataset delle etichette "*char_labels*" ha dimensioni pari a [num_images, 7, 37].

Dal momento in cui le targhe italiane possiedono sette caratteri, per ciascuna delle sette posizioni ci sono 36 caratteri possibili (26 lettere, 10 cifre), memorizzati in una codifica *One-Hot Encoded*.



```
(pythonProject) clizia@clizia-X510UNR: ~/PycharmProjects/pythonProject$ python info.py
<KeysViewHDF5 ['char_labels', 'images', 'length_labels', 'license_numbers', 'noise_SNRs_db', 'scaling_target_widths']>
<HDF5 dataset "images": shape (704000, 50, 100, 3), type "uint8">
<HDF5 dataset "length_labels": shape (704000,), type "<18>">
<HDF5 dataset "char_labels": shape (704000, 7, 37), type "<f8>">
<HDF5 dataset "license_numbers": shape (704000,), type "[0]">
<HDF5 dataset "noise_SNRs_db": shape (704000,), type "<f8>">
(pythonProject) clizia@clizia-X510UNR: ~/PycharmProjects/pythonProject$
```

Figura 5.3 – Informazioni Dataset

Per ciò che concerne l'architettura della rete, è stata utilizzata la rete descritta nel paragrafo 2.3 e precisamente in figura 2.4.

Tutti i livelli convoluzionali usano la funzione di attivazione ReLU, ad eccezione del secondo livello convolutivo che utilizza una funzione di attivazione Leaky ReLU . Il dropout viene applicato dopo la funzione di attivazione dei primi due strati completamente connessi e la probabilità di *dropout* è 0.5.

Per l'addestramento della rete è stata utilizzata una dimensione di *batch* pari a 512 con learning rate 0.01, *patience* pari a 100 ed il *training* è stato interrotto

dopo che la precisione sul set di convalida non era migliorata per 1000 interazioni. La rete ha impiegato 90 ore per effettuare il *training*.

Il secondo esperimento che è stato effettuato consiste nel simulare situazioni in cui l'immagine della targa sia soggetta a sfocatura causata da movimento. Questo rappresenta uno dei casi più frequente che un esperto forense si ritrova a dover affrontare.

Pertanto, tramite l'utilizzo della libreria *imgaug* è stato aggiunto in un primo momento del rumore impulsivo tramite la funzione *CoarseDropout* con una frequenza di rumore del 10%, 30%, e 50% dei pixel dell'immagine e successivamente le immagini sono state sfocate in modo tale da simulare i movimenti della fotocamera.

Per la sfocatura è stata utilizzata la funzione *MotionBlur* utilizzando tre dimensioni di kernel diversi per avere differenti gradazioni di sfocatura: 5×5 , 7×7 , 15×15 .



	10%	30%	50%
5×5			
7×7			



Figura 5.4 – Rumore impulsivo e MotionBlur

Partendo dalle 22.000 immagini ridimensionate a 100×50 pixel, sono state ottenute un totale di 286.000 immagini che sono state suddivise in *training set*, *validation set* e *testing set* rispettivamente con una percentuale del 80, 10, 10 per cento.

Successivamente i *dataset* di *validation*, *training* e *testing* sono stati archiviati in formato HDF5.

L'architettura della rete utilizzata per l'addestramento è descritta nel paragrafo 2.3 e sono stati utilizzate le funzioni di attivazione ReLU per tutti i livelli convolutivi, ad eccezione del secondo e sesto livello convolutivo che utilizzano una funzione di attivazione Sigmoidale.

Per quanto riguarda la dimensione di batch è stata settata a 32.

Il *training* è stato interrotto dopo che la precisione sul set di convalida non era migliorata per mille interazioni.

Per effettuare l'addestramento la rete ha impiegato 26 ore.

5.2 Risultati sperimentali

Si è ritenuto opportuno valutare l'accuratezza del modello sui singoli caratteri della targa assegnando ad ognuno di essi una precisione da Top-1 a Top-5 dove Top-1 corrisponde al caso in cui il carattere corretto è il carattere predetto più probabile.

Questo tipo di classificazione risulta particolarmente utile nei casi in cui alcuni caratteri presentino similarità.

Di seguito vengono riportati i risultati ottenuti analizzando il comportamento della rete in un primo momento sulle immagini originali alle quali non sono state apportate modifiche, successivamente sul *dataset* con l'aggiunta di distorsione prospettica e rumore ed infine sul *dataset* a cui è stato applicato rumore e sfocatura da movimento.

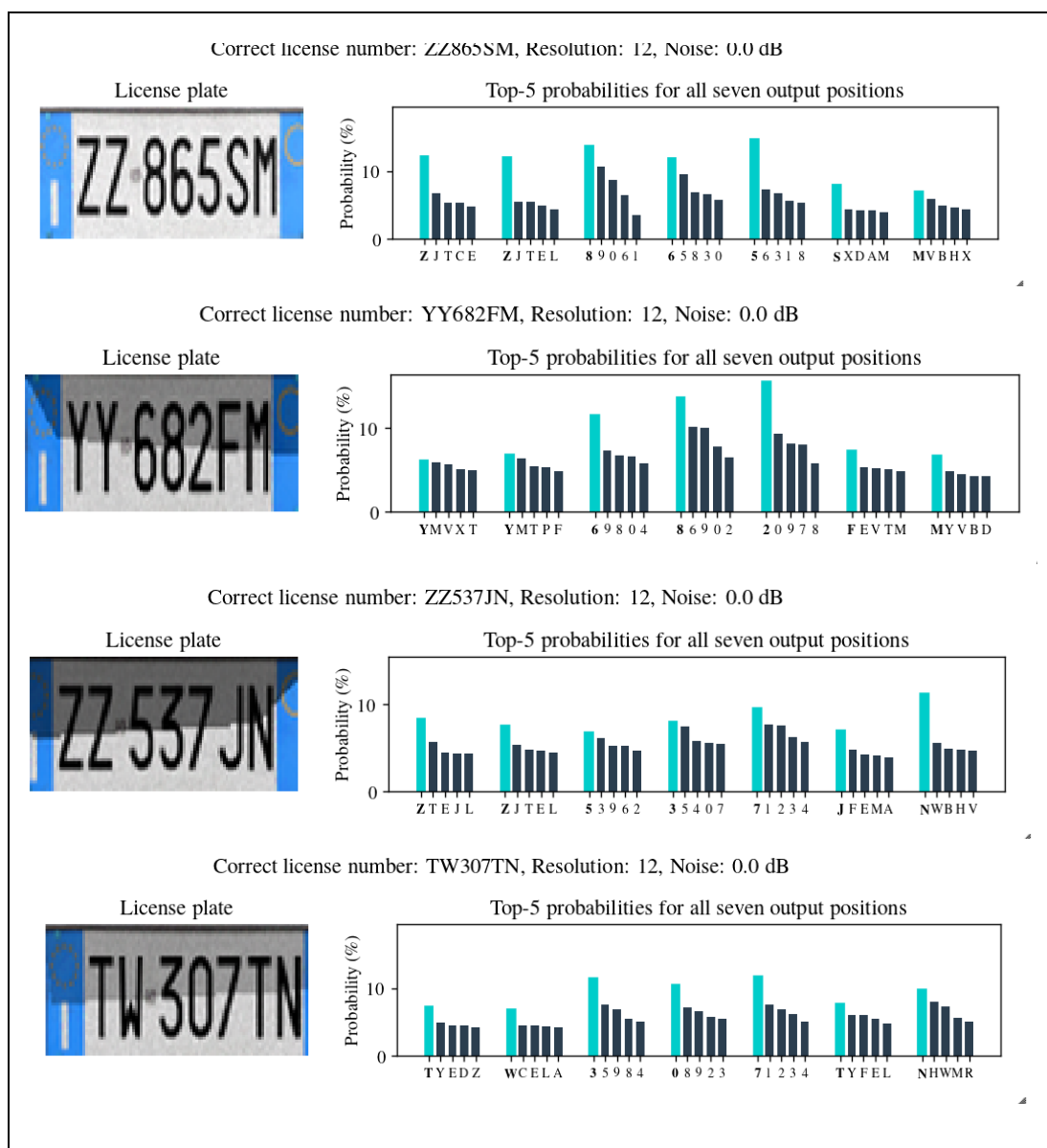


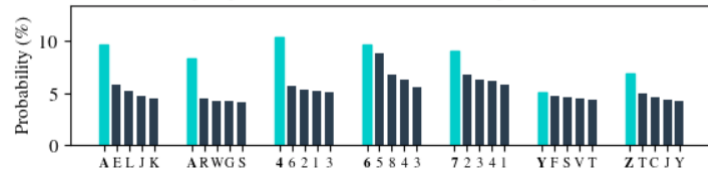
Figura 5.5 – Risultati su Dataset senza modifiche

Correct license number: AA467YZ, Resolution: 7, Noise: 0.0 dB

License plate



Top-5 probabilities for all seven output positions

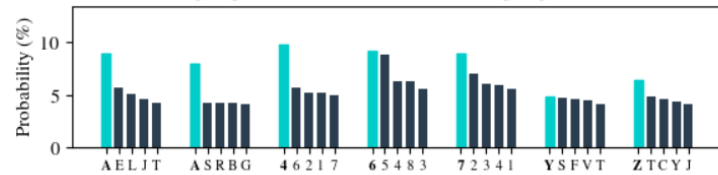


Correct license number: AA467YZ, Resolution: 7, Noise: 10.0 dB

License plate



Top-5 probabilities for all seven output positions

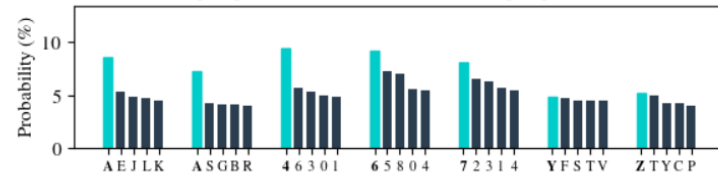


Correct license number: AA467YZ, Resolution: 7, Noise: 30.0 dB

License plate



Top-5 probabilities for all seven output positions

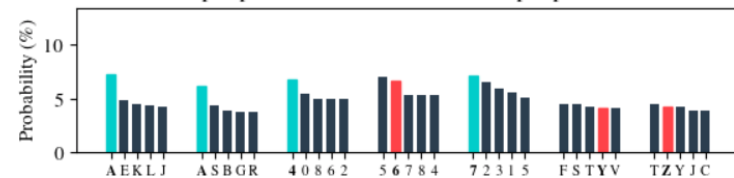


Correct license number: AA467YZ, Resolution: 7, Noise: 50.0 dB

License plate



Top-5 probabilities for all seven output positions



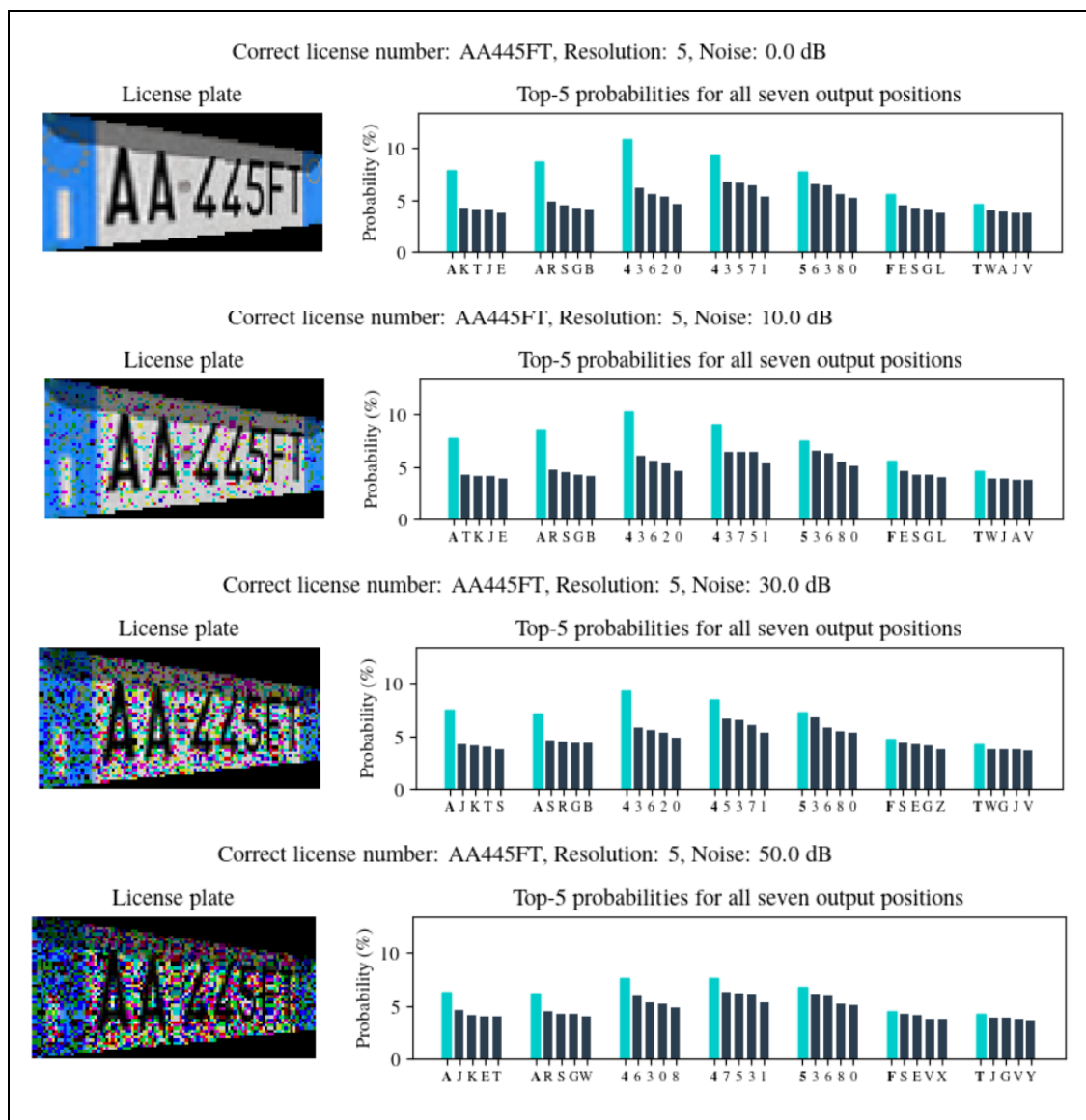


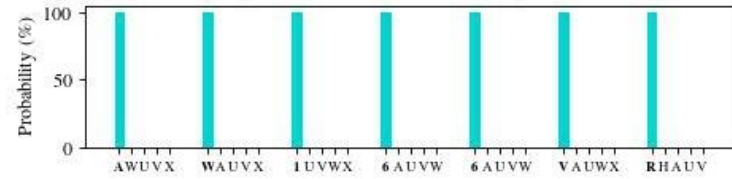
Figura 5.6 – Risultati su Dataset con distorsione prospettica e rumore

Correct license number: AW166VR, Resolution: 5, Noise: 10.0 dB

License plate



Top-5 probabilities for all seven output positions

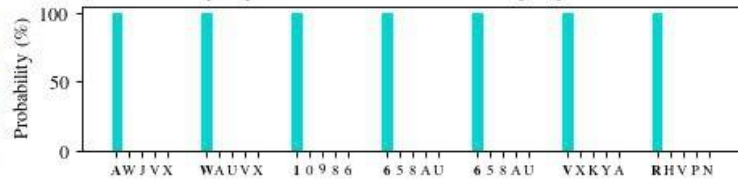


Correct license number: AW166VR, Resolution: 5, Noise: 50.0 dB

License plate



Top-5 probabilities for all seven output positions

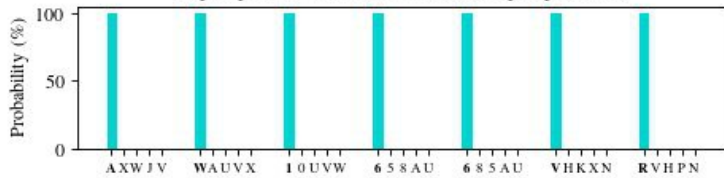


Correct license number: AW166VR, Resolution: 7, Noise: 30.0 dB

License plate



Top-5 probabilities for all seven output positions

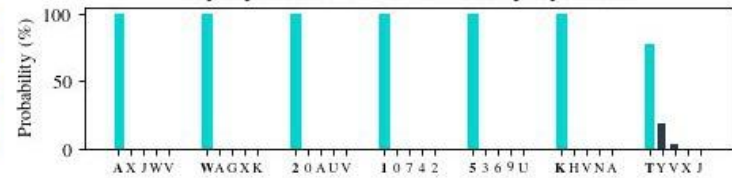


Correct license number: AW215KT, Resolution: 15, Noise: 10.0 dB

License plate



Top-5 probabilities for all seven output positions

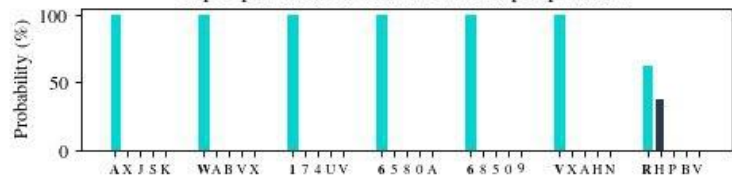


Correct license number: AW166VR, Resolution: 15, Noise: 10.0 dB

License plate



Top-5 probabilities for all seven output positions

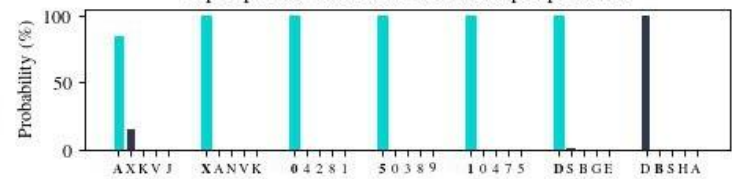


Correct license number: AX051DB, Resolution: 15, Noise: 10.0 dB

License plate



Top-5 probabilities for all seven output positions



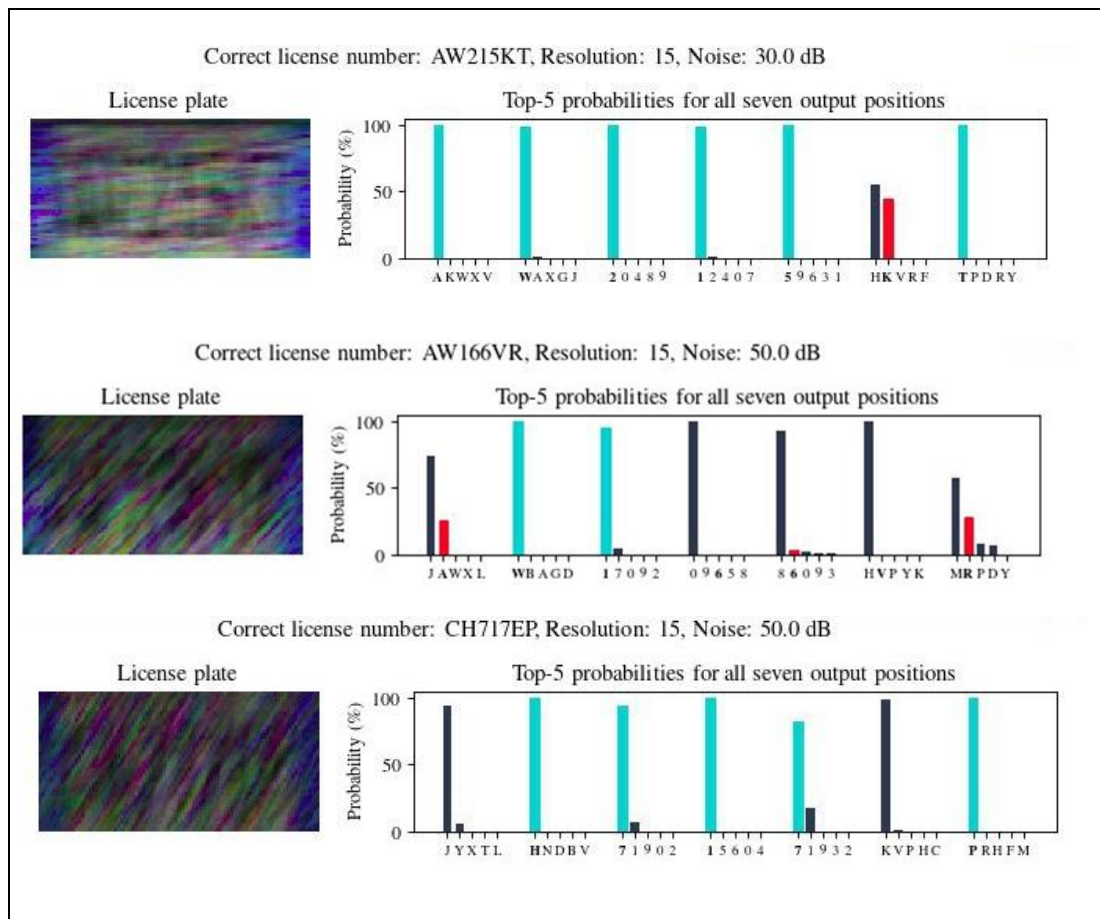


Figura 5.7 – Risultati su Dataset con rumore e MotionBlur

Dai risultati ottenuti in figura 5.6 e 5.5 è possibile verificare come le tre posizioni dei caratteri centrali abbiano probabilità più alta.

Possiamo supporre che questo sia dovuto al fatto che la rete abbia appreso che vi siano da confrontare solamente dieci cifre rispetto alle ventisei lettere delle altre posizioni e pertanto, risulti più precisa nell'identificazione dei numeri.

Infine, come si evince in figura 5.7 la rete trova qualche difficoltà nel riconoscimento dei caratteri delle targhe in cui è stata aggiunta sfocatura con un kernel di dimensioni pari a 15×15 e rumore pari al 50% dei pixel.

5.3 Conclusioni e sviluppi futuri

Nell'ambito di questa relazione di tesi è stato effettuato uno studio sul riconoscimento automatico delle targhe ad opera di algoritmi di *deep learning* quali le reti neurali.

Si è illustrato come l'utilizzo di tecniche di *deep learning* e di approcci basati sull'apprendimento diano risultati efficaci nei casi di targhe fortemente degradate e con una bassa risoluzione superando di gran lunga i risultati ottenuti dai tradizionali sistemi ANPR.

Nel dettaglio sono stati studiati diversi approcci che hanno applicazioni, finalità ed architetture di rete differenti.

Una volta analizzati i vari approcci si è ritenuto opportuno introdurre un nuovo contesto in cui l'esperto forense si ritrovi a dover affrontare immagini che presentino una forte distorsione prospettica, rumore e sfocatura causata da movimento del veicolo in questione.

A tal fine, sono state esposte le procedure preliminari per la creazione di due *dataset*, il primo a cui è stato aggiunto rumore impulsivo e distorsione prospettica e un secondo a cui è stato aggiunto rumore impulsivo e sfocatura da movimento.

Si è potuto osservare come la rete riscontri diverse difficoltà nel riconoscimento dei caratteri con il *dataset* con distorsione prospettica, in quanto la rete CNN in questione è estremamente posizionale e pertanto, tenga in considerazione della posizione dei sette caratteri della targa.

Infatti, è possibile verificare come le tre posizioni dei caratteri centrali abbiano probabilità più alta in quanto la rete ha appreso che in tale posizioni siano presenti solamente numeri ed effettua il riconoscimento basandosi esclusivamente su dieci cifre.

Per risolvere tale problematica risulta quindi necessario, tramite algoritmi di *Image Enhancement*, effettuare l'estrazione e la rotazione della targa prima di procedere al riconoscimento dei singoli caratteri.

Difatti, una volta analizzata la seguente criticità della rete si è ritenuto opportuno creare e addestrare la rete su un *dataset* con immagini fortemente degradate con rumore e *MotionBlur* a cui non è stata aggiunta alcuna distorsione prospettica.

La rete, infatti, ha dato ottimi risultati riuscendo a riconoscere la targa anche nei casi in cui la targa sia fortemente degradata, con una sfocatura molto accentuata e con il 50% dei pixel a cui è stato aggiunto rumore impulsivo.

Si ritiene opportuno precisare che, negli sviluppi futuri del progetto di tesi esposto, risulti particolarmente rilevante al fine di ottenere maggiori risultati effettuare dei cambiamenti all'architettura di rete per svincolare il riconoscimento dei singoli caratteri dalla posizione degli stessi.

Gli sviluppi futuri che si potranno implementare arricchiranno il panorama dei sistemi automatici di riconoscimento di targhe basati sull'apprendimento, col fine ultimo di agevolare l'identificazione dei caratteri nei casi più critici ottenendo risultati più accurati ed aiutare l'esperto forense durante le indagini.

Capitolo 6

Ringraziamenti

Desidero innanzitutto ringraziare il Professore Battiato per avermi guidato e supportato nella fase più importante del mio percorso accademico.

Ringrazio il dott. Oliver Giudice per avermi seguito con esperienza nel mio lavoro di tesi e per la disponibilità dimostratomi.

Un ringraziamento speciale va ai miei genitori, Melania e Daniele, che mi hanno sostenuto nel mio cammino, per essermi stati sempre accanto nei momenti di felicità e soprattutto nei momenti di preoccupazione e sconforto, per aver creduto in me e per avermi donato tutti i loro insegnamenti senza i quali non sarei ciò che sono, vi devo molto.

Ringrazio mio fratello Fabio per essere sempre al mio fianco ed appoggiare le mie scelte condivisibili e non, per essere sempre pronto ad ascoltarmi ad ogni ora del giorno nonostante i chilometri che ci separano.

Un ringraziamento va a mio zio Renato, per avermi fatta avvicinare a questo mondo sin da piccola con amore e saggezza e per tutti gli infiniti consigli.

Grazie infinite a Marco per le tante risate in questi anni, per aver sempre creduto in me e soprattutto per avermi aiutato nonostante la mia immensa caparbia.

Un ringraziamento speciale va a Francesca, collega, coinquilina ma soprattutto amica, grazie per avermi insegnato il valore dell'amicizia con dolcezza e allegria, senza di te non sarei arrivata fino a qui.

Grazie a Matteo, collega e amico con cui ho condiviso i momenti più belli di questi tre anni, per le tante risate, per tutte le sveglie alle 7 del mattino e per tutte le volte che mi hai dovuto aspettare sotto casa.

Un grazie speciale va a Flavia, amica d'infanzia e compagna di mille avventure con cui ho condiviso molte delle esperienze più belle della mia vita, grazie per esserci sempre.

A Martina, Luca, Jeremy e Davide per aver condiviso gioie e fatiche di questi tre anni trascorsi insieme.

Infine, ringrazio Claudia, coinquilina e amica che Catania mi ha donato, porterò nel mio cuore i momenti più belli e divertenti di questi tre anni in casa Leonardi, Bob rimarrà sempre una parte di me.

Dedico questo traguardo a mia zia Rosetta che avrebbe voluto condividere con me questo giorno tanto atteso, spero di averti resa orgogliosa di me.

Riferimenti

- [1] **Acquisizione e analisi forense di sistemi di videosorveglianza**
Sebastiano Battiato, Antonmarco Catania, Fausto Galvan, Martino Jerian, Ludovico Paveri Fontana
IISFA Memberbook – 2015
- [2] **Forensic Reconstruction of Severely Degraded License Plates**
Benedikt Lorch, Shruti Agarwal, Hany Farid
Society for Imaging Science & Technology (Eds.), Electronic Imaging. Burlingame, CA, US – 2019
- [3] **Learning to Decipher License Plates in Severely Degraded Images**
Paula Kaiser, Franziska Schirmacher, Benedikt Lorch, Christian Riess
Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part VI. Springer International Publishing, 2021
- [4] **Neural Network for Denoising and Reading Degraded License Plates**
Gianmaria Rossi, Marco Fontani, Simone Milani
Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part VI. Springer International Publishing, 2021
- [5] **Holistic recognition of low quality license plates by CNN using track annotated data**
Jakub Spanhel, Jakub Sochor, Roman Juranek, Adam Herout, Lukas Marsik, Pavel Zemcik
14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2017
- [6] **Complex wavelet structural similarity: A new image similarity index**
Mehul P. Sampat, Zhou Wang, Shalini Gupta, Alan Conrad Bovik, Mia K. Markey
Society for Imaging Science & Technology (Eds.), Electronic Imaging. Burlingame, CA, US – 2019
- [7] **History of ANPR**
<http://www.anpr-international.com/history-of-anpr/>