Università degli Studi di Salerno

Corso di Laurea Informatica

Fondamenti di Intelligenza Artificiale

SignVision

Autori:

Iari Normanno (Matricola: 0512116162)
Marco Acierno (Matricola: 0512116378)

Repository

`https://github.com/wassupiari/SignVision`

Contents

1 Introduzione

L'intelligenza artificiale sta rivoluzionando numerosi settori attraverso l'analisi e il riconoscimento delle immagini. Uno dei campi di maggiore applicazione riguarda la **computer vision**, ossia la capacità delle macchine di interpretare il contenuto delle immagini e di estrarre informazioni significative.

Il riconoscimento automatico dei segnali stradali è un problema rilevante nel campo della sicurezza stradale e dell'assistenza alla guida. **SignVision** è un progetto di machine learning basato su reti neurali convoluzionali (CNN) per il riconoscimento automatico dei segnali stradali presenti in immagini statiche.

1.1 Obiettivi del sistema

Il sistema SignVision ha l'obiettivo di riconoscere e classificare automaticamente i segnali stradali a partire da immagini statiche. Analizzando le caratteristiche visive dei segnali, il modello è in grado di associarli alla categoria corretta con un'elevata accuratezza.

1.2 Struttura Documentazione

La documentazione di **SignVision** è organizzata in diverse sezioni per fornire una visione chiara e completa del progetto. Di seguito, vengono descritte le principali sezioni:

- **Introduzione:** Viene presentata una panoramica del problema affrontato, gli obiettivi del progetto e la motivazione alla base della sua realizzazione.
- **Analisi del problema:** Include la specifica PEAS, le proprietà dell'ambiente in cui opera il sistema e una descrizione degli attuatori e dei sensori utilizzati.
- **Dataset:** Descrizione dettagliata del dataset utilizzato, compresa la raccolta dei dati, la loro organizzazione e le trasformazioni applicate per il pre-processing.
- **Soluzione Proposta:** Descrizione del modello sviluppato, della sua architettura, delle tecnologie utilizzate e delle tecniche adottate per l'addestramento e l'ottimizzazione.

- **Analisi delle Epoche di Addestramento:** Esposizione dei risultati ottenuti durante l'addestramento del modello con configurazioni diverse di epoche e scelta del modello ottimale.
- **Conclusioni:** Una riflessione sui risultati ottenuti, le possibili applicazioni future e i miglioramenti che possono essere implementati per ottimizzare il sistema.

2 Analisi del problema

2.1 Specifica PEAS

2.1.1 Performance

La misura di performance del sistema si basa sulla capacità di SignVision di classificare correttamente i segnali stradali a partire da immagini statiche. L'accuratezza della predizione è il principale criterio di valutazione, con particolare attenzione alla corretta identificazione delle diverse categorie di segnali.

2.1.2 Environment

L'ambiente in cui opera SignVision è costituito da immagini digitali di segnali stradali, provenienti da dataset strutturati o caricate dagli utenti.

2.1.3 Actuators

Gli attuatori del sistema comprendono l'output della classificazione del segnale stradale, che viene restituito attraverso un'interfaccia grafica. Gli utenti possono caricare un'immagine e visualizzare immediatamente la classe assegnata dal modello, verificando il risultato della classificazione.

2.1.4 Sensors

I sensori del sistema sono costituiti dalle immagini fornite dagli utenti o dai dataset.

2.2 Proprietà dell'ambiente

L'ambiente in cui opera il sistema SignVision è caratterizzato da una serie di proprietà che ne influenzano il funzionamento e la capacità di riconoscere correttamente i segnali stradali nelle immagini. Queste proprietà possono essere descritte nei seguenti termini:

- **Completamente osservabile**: L'ambiente è completamente osservabile, poiché ogni immagine contiene tutte le informazioni necessarie per determinare la classe del segnale stradale senza richiedere ulteriori dati esterni.
- **Stocastico**: L'ambiente è stocastico, poiché la classificazione di un segnale può essere influenzata da fattori come illuminazione, angolazione,

qualità dell'immagine o distorsioni. Il modello può assegnare probabilità diverse alle classi in base alle caratteristiche dell'immagine, e lo stesso segnale potrebbe essere classificato diversamente in condizioni leggermente differenti.

- **Episodico**: L'ambiente è episodico, in quanto ogni immagine viene analizzata indipendentemente dalle altre. La classificazione di un segnale non ha effetto sulle predizioni successive, e ogni input rappresenta un caso isolato.

- **Statico**: L'ambiente è statico, poiché le immagini non cambiano nel tempo una volta acquisite. Il modello lavora su un input fisso, senza aggiornamenti dinamici durante l'analisi.

- **Discreto**: L'ambiente è discreto, in quanto i segnali stradali appartengono a un numero finito di classi ben definite. Il sistema deve assegnare ogni immagine a una delle categorie predefinite, senza gestire dati continui o classi indefinite.

- **Singolo**: Il sistema opera come un agente singolo, interagendo con l'ambiente in modo indipendente, senza la necessità di coordinarsi con altri sistemi intelligenti per prendere decisioni.

3 Dataset

3.1 Raccolta dei dati

Il dataset utilizzato per l'addestramento e la valutazione del sistema SignVision è il **German Traffic Sign Recognition Benchmark** (GTSRB), un dataset ampiamente utilizzato nel campo della visione artificiale per il riconoscimento dei segnali stradali. Questo dataset contiene oltre **50.000 immagini** appartenenti a **43 classi diverse** di segnali stradali raccolte in vari contesti, con differenze in termini di illuminazione, angolazione e qualità dell'immagine.

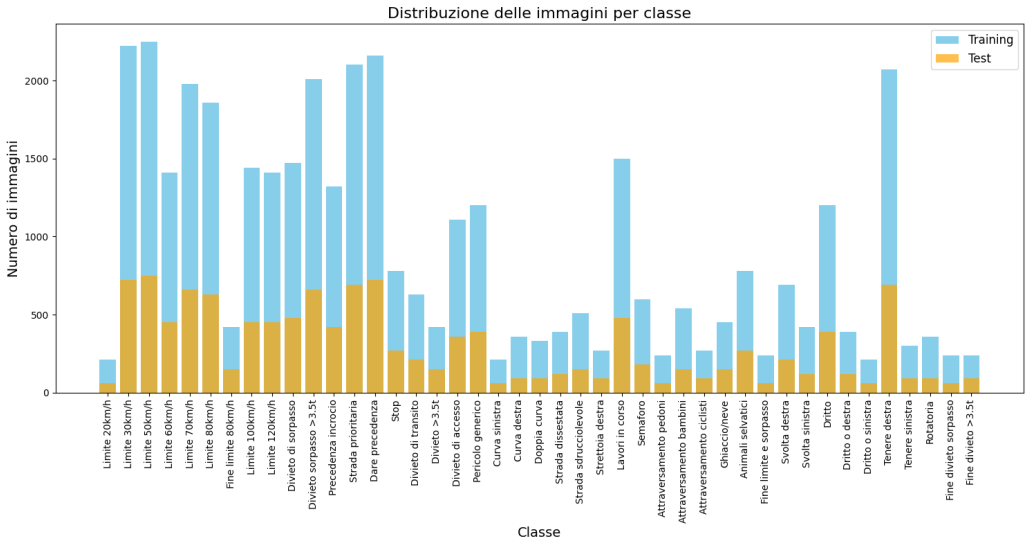


Figure 1: Distribuzione delle immagini tra classi nel train/test set.

I dati sono suddivisi in due insiemi principali:

- **Set di addestramento**: composto da immagini etichettate che vengono utilizzate per insegnare al modello a riconoscere i segnali stradali.
- **Set di test**: contiene immagini non viste durante l'addestramento, utilizzate per valutare l'accuratezza e la capacità di generalizzazione del modello.

Ogni immagine nel dataset è accompagnata da una classe associata, rappresentata da un'etichetta numerica che identifica il tipo di segnale stradale presente. I dati sono organizzati in un formato strutturato, con un file CSV che elenca i percorsi delle immagini e le relative etichette.

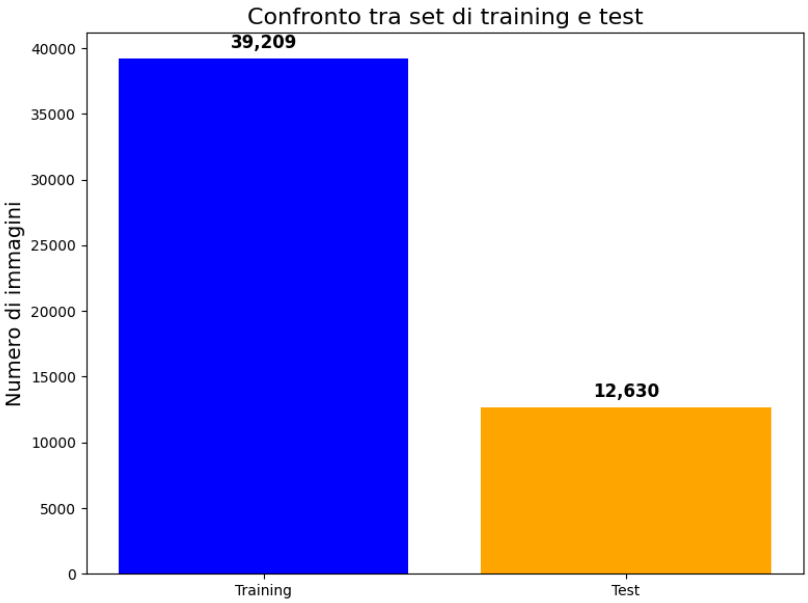


Figure 2: Confronto tra train e test set.

Una caratteristica del dataset GTSRB è che tutte le immagini sono in formato **30x30 pixel**. Questo ha reso necessario lo sviluppo di una **rete neurale convoluzionale (CNN) da zero**, poiché i modelli pre-addestrati più diffusi sono stati sviluppati per immagini di dimensioni molto maggiori e non risultavano adeguati per questo tipo di input.

4 Soluzione Proposta

4.1 Soluzione Proposta

La soluzione proposta è un sistema di Machine Learning progettato per la classificazione di immagini, rilevando la presenza di specifiche caratteristiche indicative di un determinato fenomeno. Nel nostro caso, il modello analizza le immagini per riconoscere i vari segnali.

4.2 Modello

In questa sezione, verrà descritta l'architettura del modello proposto insieme alle sue principali caratteristiche.

4.2.1 Tecnologie del modello

Il modello scelto si basa su una rete neurale convoluzionale(Convolution Neural Network - CNN), che è particolarmente efficace per la classificazione di immagini. Le CNN operano attraverso una serie di trasformazioni sui dati in input, identificando progressivamente caratteristiche rilevanti come bordi, forme e strutture più complesse. Il modello è stato implementato utilizzando Python 3.9 e alcune librerie:

- **TensorFlow/Keras:** per la costruzione, l'addestramento e il testing del modello di rete neurale.
- **NumPy:** per operazioni numeriche ad alte prestazioni.

4.2.2 Rete neurale

Il modello di rete neurale convoluzionale (CNN) sviluppato per il riconoscimento dei segnali stradali è composto da diversi componenti chiave, progettati per massimizzare l'accuratezza riducendo il rischio di overfitting. L'architettura include i seguenti componenti:

- **Input Layer:** Il modello accetta immagini in input con dimensioni **30×30 pixel e 3 canali RGB**.
- **Primo blocco convoluzionale:**
  - **Conv2D (32 filtri, kernel 5x5, ReLU)**: estrae le caratteristiche di base dai segnali stradali.
  - **BatchNormalization**: migliora la stabilità dell'addestramento.

  - **Conv2D (64 filtri, kernel 3x3, ReLU)**: apprende pattern più complessi.
  - **BatchNormalization**: aiuta la convergenza del modello.
  - **MaxPooling2D (2x2)**: riduce la dimensionalità delle feature maps.
  - **Dropout (15%)**: previene l'overfitting.

- **Secondo blocco convoluzionale:**
  - **Conv2D (128 filtri, kernel 3x3, ReLU)**: rileva pattern più avanzati.
  - **BatchNormalization**
  - **MaxPooling2D (2x2)**
  - **Dropout (25%)**

- **Livello Fully Connected (Denso):**
  - **Flatten**: converte le feature maps in un vettore 1D.
  - **Dense (256 neuroni, ReLU)**: livello completamente connesso per l'apprendimento delle rappresentazioni.
  - **BatchNormalization**
  - **Dropout (50%)**
  - **Dense (43 neuroni, Softmax)**: livello di output per classificare i segnali stradali.

- **Compilazione del Modello:**
  - Ottimizzatore: **Adam (learning rate = 0.001)**
  - Funzione di perdita: **Categorical Crossentropy**
  - Metriche: **Accuracy**

4.2.3 Funzione di attivazione

Le funzioni di attivazione giocano un ruolo essenziale nel modello, consentendo alla rete di introdurre non-linearità e di apprendere rappresentazioni più complesse dei dati. Nel modello implementato per SignVision, sono state utilizzate due funzioni di attivazione principali:

- **ReLU (Rectified Linear Unit)**: Viene impiegata nei livelli convoluzionali e nei livelli densi intermedi. Questa funzione attiva un neurone solo se il valore in ingresso è positivo, altrimenti restituisce zero. L'uso di ReLU permette alla rete di apprendere più rapidamente e di mitigare il problema della scomparsa del gradiente, comune nelle reti profonde.
- **Softmax**: È utilizzata nel livello di output per trasformare le attivazioni della rete in una distribuzione di probabilità sulle 43 classi di segnali stradali. Softmax assegna un punteggio probabilistico a ciascuna classe, garantendo che la somma totale delle probabilità sia pari a 1. Questo consente di identificare la classe più probabile e di valutare il livello di incertezza nella previsione.

4.2.4 Funzione di ottimizzazione

L'ottimizzazione è una fase cruciale nel processo di addestramento della rete neurale, poiché determina l'aggiornamento dei pesi per migliorare le prestazioni del modello. Per il modello **SignVision**, è stato scelto l'ottimizzatore **Adam** (Adaptive Moment Estimation), una delle tecniche più efficaci e ampiamente utilizzate per il training delle reti neurali.

Nel modello implementato, è stato utilizzato un learning rate predefinito di 0.001, un valore comunemente adottato per garantire un buon equilibrio tra velocità di convergenza e stabilità dell'addestramento. L'ottimizzatore Adam è stato scelto per la sua capacità di gestire problemi complessi di classificazione come quello dei segnali stradali, permettendo di ottenere una buona accuratezza senza richiedere un tuning eccessivo dei parametri.

4.2.5 Funzione di perdita

Per questo problema di classificazione multiclasse è stata scelta la funzione di perdita **Categorical Cross-Entropy**, ideale per reti neurali che devono assegnare un' osservazione a una delle 43 classi disponibili.

4.3 Dataset Training

Il processo di training del modello si è svolto utilizzando il dataset **German Traffic Sign Recognition Benchmark** (GTSRB), composto da immagini di segnali stradali classificati in 43 classi diverse. Il dataset è stato suddiviso in tre insiemi principali: **training**, **validation** e **test**.

- **Training set:** 80% delle immagini totali, utilizzate per addestrare la rete neurale, migliorando la capacità di generalizzazione.

- **Validation set:** 20% delle immagini, utilizzate per monitorare le prestazioni del modello durante il training e prevenire l'overfitting.
- **Test set:** Un insieme separato di immagini indipendenti, usato per la valutazione finale del modello.

Tutte le immagini sono state ridimensionate a una risoluzione di 30x30 pixel e normalizzate per migliorare la stabilità numerica durante l'addestramento.

4.3.1 Bilanciamento delle classi

Poiché alcune delle 43 classi del dataset presentano un numero di immagini significativamente inferiore rispetto ad altre (dataset sbilanciato), abbiamo applicato dei **pesi di classe** (*class weights*) in fase di addestramento. In questo modo, gli errori di classificazione sulle classi meno rappresentate incidono maggiormente sulla funzione di perdita, aiutando il modello a non trascurare le categorie minori e a migliorare le performance su classi scarsamente rappresentate.

4.4 Strategia di addestramento

Durante la fase di training, abbiamo sperimentato con diverse configurazioni per ottimizzare le prestazioni del modello. Nello specifico, abbiamo testato il modello con **5, 15 e 20 epoche** per determinare il numero ottimale di iterazioni necessarie a raggiungere un buon equilibrio tra accuratezza e overfitting.

4.4.1 Data Augmentation

Un aspetto fondamentale per migliorare la capacità di generalizzazione del modello è stato l'utilizzo della **data augmentation**, una tecnica che permette di aumentare artificialmente la varietà del dataset senza acquisire nuove immagini. Questo è particolarmente utile in problemi di classificazione di immagini, in cui un dataset limitato potrebbe portare il modello a sovradattarsi ai dati di training.

Applicando trasformazioni geometriche e fotometriche alle immagini di addestramento, il modello è stato esposto a variazioni realistiche che potrebbe incontrare nel mondo reale, come variazioni di angolazione, illuminazione o distorsione prospettica. Le seguenti trasformazioni sono state utilizzate per aumentare la robustezza del modello:

- **Rotazione** fino a 15°

- **Traslazioni orizzontali e verticali** fino al 20%
- **Shear Transformation** fino al 10%
- **Zoom** fino al 10%
- **Fill Mode:** *nearest* per completare eventuali aree vuote dopo le trasformazioni

L'uso della **data augmentation** ha permesso di migliorare la capacità del modello di riconoscere segnali stradali indipendentemente da piccoli cambiamenti nella posizione, nella scala o nell'orientazione. Questo ha ridotto significativamente il rischio di overfitting, consentendo al modello di apprendere caratteristiche più generali piuttosto che memorizzare i dettagli delle immagini specifiche del training set.

4.4.2 Meccanismi di regolazione del training

Per migliorare la stabilità dell'addestramento e prevenire sia il sovradattamento che la divergenza del modello, sono stati adottati i seguenti accorgimenti:

Early Stopping Per evitare il sovradattamento e ridurre i tempi di training, è stato implementato un **callback di Early Stopping**. Questo meccanismo monitora la funzione di perdita sul validation set (*val_loss*) e interrompe automaticamente l'addestramento se il modello smette di migliorare per un numero consecutivo di epoche.

Il callback è stato configurato come segue:

- **Monitor**: *val_loss* - interrompe il training quando l'errore sul validation set non migliora.
- **Pazienza**: 5 epoche - se la funzione di perdita non migliora per 5 epoche consecutive, l'addestramento viene interrotto.
- **Ripristino dei migliori pesi**: *restore_best_weights=True* - al termine del training, vengono ripristinati i pesi corrispondenti alla miglior performance ottenuta sul validation set.

Questa strategia garantisce che il modello non continui ad apprendere dettagli irrilevanti dopo aver raggiunto il suo miglior punto di generalizzazione, prevenendo il rischio di sovradattamento ai dati di training.

ReduceLROnPlateau Per ottimizzare ulteriormente il processo di apprendimento, è stato impiegato il meccanismo **ReduceLROnPlateau**, che riduce automaticamente il tasso di apprendimento quando la metrica di validazione smette di migliorare. Nello specifico, è stato impostato con i seguenti parametri:

- **Monitor**: *val_loss* - la funzione monitora l'errore sul validation set.
- **Fattore di riduzione**: 0.5 - quando il miglioramento si arresta, il learning rate viene dimezzato.
- **Pazienza**: 3 epoche - la riduzione avviene solo se non si osservano miglioramenti per 3 epoche consecutive.
- **Learning rate minimo**: 1×10^{-6} - il tasso di apprendimento non può scendere sotto questa soglia.

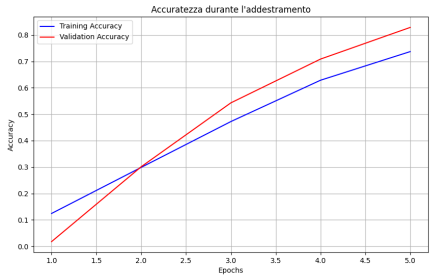
Early Stopping e **ReduceLROnPlateau** lavorano insieme per garantire un addestramento efficace: il primo impedisce il sovradattamento interrompendo il training al momento opportuno, mentre il secondo permette di adattare dinamicamente la velocità di apprendimento per migliorare la convergenza del modello.

4.5 Analisi delle Epoche di Addestramento

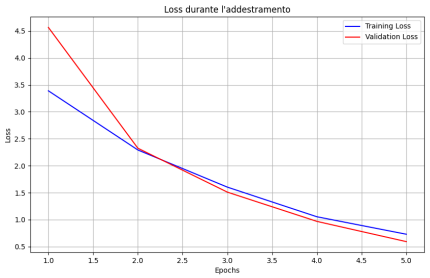
Durante il processo di addestramento del modello, sono state provate diverse configurazioni di epoche (5, 15 e 20) per identificare quella che garantisse il miglior compromesso tra accuratezza e generalizzazione. In questa fase, abbiamo esaminato l'andamento dell'accuratezza e della perdita sia per il set di addestramento che per quello di validazione.

4.5.1 5 epoche

Dai grafici sopra, si nota un miglioramento progressivo delle prestazioni del modello durante le prime 5 epoche. Tuttavia, per ottenere risultati più stabili, abbiamo ampliato il numero di epoche e valutato i miglioramenti aggiuntivi.



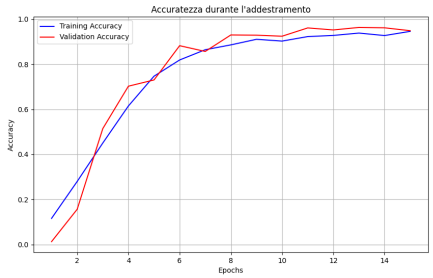
(a) Accuratezza durante il training e validazione (5 epoche)



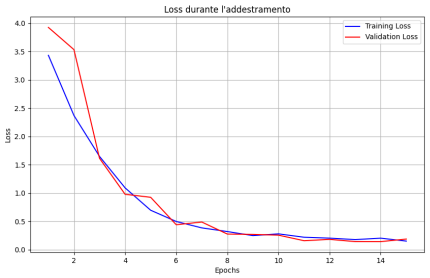
(b) Loss durante il training e validazione (5 epoche)

Figure 3: Analisi dei risultati con 5 epoche di addestramento

4.5.2 15 epoche



(a) Accuratezza durante il training e validazione (15 epoche)



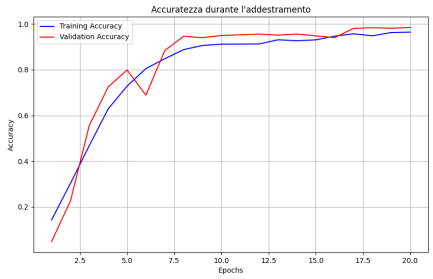
(b) Loss durante il training e validazione (15 epoche)

Figure 4: Analisi dei risultati con 15 epoche di addestramento

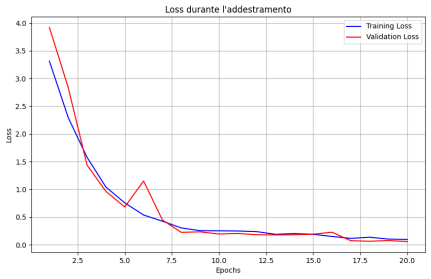
Dai grafici sopra, si osserva un miglioramento costante delle prestazioni del modello man mano che il numero di epoche aumenta. Le prime 15 epoche mostrano un andamento positivo sia per l'accuratezza che per la riduzione della perdita, suggerendo un punto promettente per la selezione finale del numero di epoche da utilizzare.

4.5.3 20 epoche

I grafici relativi alla perdita e all'accuratezza durante l'addestramento mostrano una convergenza stabile dopo circa 15 epoche, suggerendo che oltre questo punto l'aggiunta di ulteriori epoche non porta a miglioramenti significativi nelle prestazioni del modello



(a) Accuratezza durante il training e validazione (20 epoche)



(b) Loss durante il training e validazione (20 epoche)

Figure 5: Analisi dei risultati con 20 epoche di addestramento

4.6 Scelta delle Epoche Ottimali

Dopo aver analizzato i risultati delle configurazioni con 5, 15 e 20 epoche, abbiamo osservato che 15 epoche rappresentano il punto ottimale per garantire un'accurata generalizzazione del modello, evitando al contempo il rischio di overfitting. Di seguito, presentiamo i risultati finali del modello addestrato per 15 epoche:

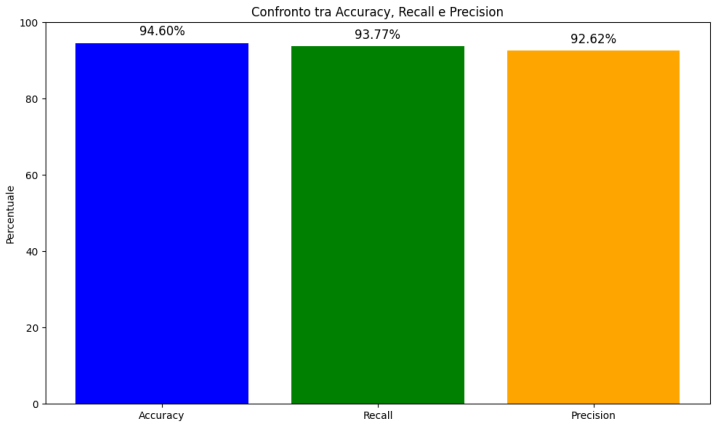


Figure 6: Confronto tra Accuracy, Recall e Precision del modello finale

La matrice di confusione seguente evidenzia le performance del modello finale sulle diverse classi di segnali stradali:

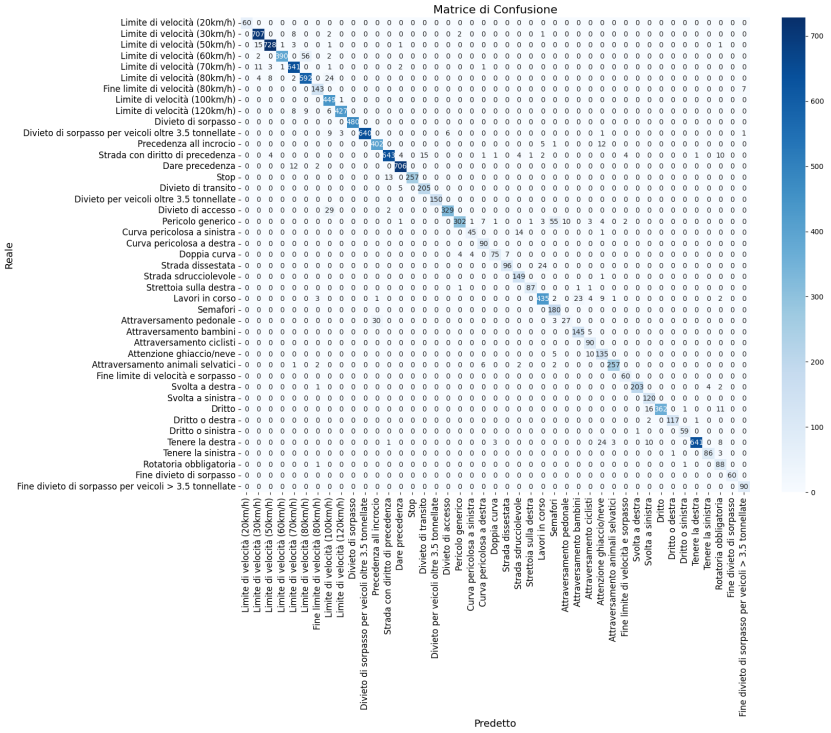


Figure 7: Matrice di Confusione del modello finale

In sintesi, l'addestramento per 15 epoche ha permesso di ottenere un'accuratezza del 94%, con metriche di recall e precision bilanciate, dimostrando una solida capacità di generalizzazione sui dati di test.

5 Conclusioni

Il progetto **SignVision** ha dimostrato come le reti neurali convoluzionali (CNN) possano essere utilizzate con successo per il riconoscimento dei segnali stradali, ottenendo risultati eccellenti sia in termini di accuratezza che di generalizzazione.

Attraverso l'uso di un dataset strutturato come il **German Traffic Sign Recognition Benchmark (GTSRB)** e l'applicazione di tecniche avanzate di addestramento, come il **Data Augmentation** e la regolazione dinamica del tasso di apprendimento, il modello ha raggiunto un'accuratezza del 94% sui dati di test, con un buon equilibrio tra recall e precision.

Nonostante i risultati ottenuti, ci sono margini di miglioramento futuri:

- **Integrazione di modelli pre-addestrati**: L'uso di reti pre-addestrate come ResNet o EfficientNet potrebbe migliorare ulteriormente le prestazioni del sistema.
- **Espansione del dataset**: L'aggiunta di nuove immagini, magari provenienti da condizioni diverse (notturne, nebbia, ecc.), aumenterebbe la capacità del modello di gestire scenari reali complessi.
- **Ottimizzazione computazionale**: Ridurre i tempi di inferenza ottimizzando il modello per dispositivi embedded o sistemi edge.
- **Implementazione in un sistema reale**: Un'integrazione in sistemi di assistenza alla guida o auto a guida autonoma rappresenta un obiettivo ambizioso ma fattibile per il futuro.

In conclusione, il sistema **SignVision** rappresenta un passo significativo verso l'automazione del riconoscimento dei segnali stradali, contribuendo a migliorare la sicurezza stradale e a facilitare lo sviluppo di veicoli autonomi. I risultati ottenuti confermano l'efficacia delle tecniche di machine learning e suggeriscono ulteriori aree di ricerca e sviluppo per perfezionare il sistema.