



**Politecnico
di Torino**

Politecnico di Torino

Ingegneria Informatica (Computer Engineering)

A.a. 2024/2025

Sessione di laurea Luglio 2025

Valutazione dello stato di salute dei Noccioli mediante UAV e Deep Learning

Relatori:

Maurizio Morisio

Candidati:

Francesco Volpi

Ringraziamenti

Prima di procedere con la trattazione, vorrei dedicare qualche riga a tutti coloro che mi sono stati vicini in questo percorso di crescita personale e professionale.

Vorrei innanzitutto ringraziare il mio relatore, Morisio Maurizio, che mi ha seguito, passo dopo passo, in questo percorso.

Un sentito grazie va ai miei genitori, per avermi dato l'opportunità di seguire i miei sogni, e a tutta la mia famiglia che non mi ha mai fatto mancare affetto e sostegno.

Un ringraziamento speciale alla mia fidanzata, Altea: grazie per il tempo che mi hai dedicato, per la pazienza e per esserci sempre stata, anche nei momenti più difficili.

Grazie ai miei amici, che in questi anni sono stati una presenza fondamentale.

Infine, desidero dedicare questo lavoro a mia nonna Paola e a mia nonna Ninetta, venute a mancare quest'anno. Porto con me il loro amore, la loro saggezza e tutto ciò che mi hanno insegnato senza bisogno di parole.

Indice

Elenco delle tabelle	VI
Elenco delle figure	VII
1 Introduzione	1
1.1 Goal	2
1.2 Struttura della tesi	2
2 Dataset	4
2.1 Descrizione generale del dataset	4
2.1.1 Dataset di Carrù	4
2.1.2 Dataset di Farigliano	5
2.2 Formato e tipologia delle immagini	7
2.3 Esplorazione del Dataset	8
3 Preprocessing	13
3.1 Metodi di segmentazione in letteratura	13
3.1.1 Segmentazione tramite combinazione di indici vegetazionali .	13
3.1.2 Metodo per ulivi e rilevamento di <i>Xylella fastidiosa</i>	14
3.1.3 Metodo "Citrus" per la segmentazione di alberi di agrumi . .	15
3.2 Implementazione dei metodi di segmentazione	17
3.2.1 Segmentazione mediante indice NDVI	18
3.2.2 Segmentazione mediante il metodo <i>Citrus</i>	21
3.3 Estrazione automatica dell'albero centrale	25
3.4 Image Subdivision	28
4 Training dei Modelli e Strategie di Deep Learning	31
4.1 Introduzione al Deep Learning	31
4.1.1 Backpropagation e Gradient Descent	32
4.1.2 Attivazione Non Lineare	32
4.1.3 Overfitting e Generalizzazione	32

4.1.4	Applicazioni del Deep Learning in Agricoltura	33
4.2	Impostazione Iniziale e Riproduzione del Lavoro Precedente	34
4.2.1	Tecniche Utilizzate: Transfer Learning e Data Augmentation	34
4.2.2	Architetture Scelte	35
4.2.3	Strategia di Addestramento	37
4.2.4	Risultati Preliminari	39
4.3	Modifiche e Prove di nuove architetture	39
4.3.1	Modifiche architetturali e strategie di training	40
4.3.2	Architetture testate	42
4.3.3	Risultati	48
4.4	Estensione all'analisi multispettrale	52
4.4.1	Adattamenti al setup di training per immagini multispettrali	53
5	Conclusioni e Sviluppi Futuri	58
	Bibliografia	60

Elenco delle tabelle

4.1	Metriche di performance per ciascun modello	51
4.2	Prestazioni dei modelli nella classificazione di immagini multispettrali. Precision e recall si riferiscono alla classe “malata”.	56

Elenco delle figure

2.1	Veduta aerea del campo di noccioli a Carrù.	5
2.2	Esempi di immagini acquisite nel campo di Carrù in tre differenti giornate.	5
2.3	Veduta aerea del campo di noccioli a Farigliano.	6
2.4	Esempi di immagini acquisite nel campo di Farigliano in tre differenti giornate.	6
2.5	Esempio delle sei immagini acquisite per lo stesso albero	8
2.6	Esempio di immagine contenente più alberi	9
2.7	Esempio di immagine con alberi sovrapposti	9
2.8	La stessa pianta ripresa con differenti condizioni di illuminazione . .	10
2.9	Esempio di immagine in cui l'albero non è centrato	11
2.10	Evidenza visiva del disallineamento tra immagine RGB e immagine NIR. La parte inferiore mostra uno zoom sull'area evidenziata nella scena principale, mettendo in luce lo shift tra le due acquisizioni causato dalla diversa posizione delle ottiche.	12
3.1	Esempi di segmentazione e classificazione	14
3.2	Pipeline di segmentazione e classificazione dello stato di salute per ulivi.	15
3.3	Schema del metodo proposto	17
3.4	Confronto tra GNDVI calcolato con e senza allineamento delle bande. Il disallineamento tra la banda verde e la NIR genera artefatti visibili nei contorni degli alberi. L'allineamento migliora la qualità della segmentazione.	19
3.5	Workflow del metodo: partendo dall'immagine RGB, si calcola la mappa NDVI, si estraggono i contorni delle aree vegetate mediante threshold e si filtrano per ottenere solo quelli rilevanti.	20
3.6	Esempi di segmentazione NDVI in diversi scenari. I primi due mostrano risultati positivi, con alcune eccezioni nei dettagli. Gli ultimi due evidenziano l'importanza della soglia scelta per un corretto bilanciamento tra accuratezza e inclusività.	21

3.7	Schema del workflow completo del metodo Citrus. Le componenti contrassegnate con una X rossa indicano le parti non implementate nel presente lavoro.	22
3.8	Esempio completo del workflow applicato a una singola immagine RGB. Le fasi mostrate comprendono: (a) immagine RGB originale, (b) conversione in spazio HSI ed equalizzazione selettiva della componente di intensità, (c) generazione della mappa cromatica G-R, (d) sogliatura con algoritmo di Otsu, (e) filtraggio morfologico delle ROI, (f) visualizzazione dei contorni degli alberi sovrapposti all'immagine RGB.	24
3.9	Confronto qualitativo dei risultati ottenuti con il metodo Citrus: due esempi riusciti (in alto) e due falliti (in basso).	25
3.10	Esempi di estrazione automatica dell'albero centrale tramite sliding window. A sinistra un caso in cui l'albero è ben separato e i contorni risultano chiari. A destra un caso più complesso in cui gli alberi risultano sovrapposti.	26
3.11	Mappatura dei campi mediante metodo semi-automatico GPS + interpolazione.	27
3.12	Esempio illustrativo relativo al secondo metodo di estrazione automatica dell'albero centrale.	28
3.13	Esempi di suddivisione dell'immagine segmentata in griglie $N \times N$, in funzione dell'area occupata dalla chioma. Si noti come il valore di N cresca con l'aumentare della dimensione dell'albero.	29
4.1	Schema concettuale di una rete neurale [6]	31
4.2	Esempio di overfitting, underfitting e buona generalizzazione [7]	33
4.3	Esempio di <i>data augmentation</i> : immagine originale e sue trasformazioni [8].	35
4.4	Struttura dell'architettura ResNet50.	36
4.5	Struttura dell'architettura DenseNet121.	37
4.6	Architetture di VGG16 e VGG19 [9].	42
4.7	Modello VGG16 trainato	43
4.8	Modello VGG19 trainato	43
4.9	Schema di un Inception block [10].	44
4.10	Esempio di un residual block.	44
4.11	Modello Inception trainato.	45
4.12	Modello ResNet trainato.	45
4.13	(a) rappresenta una rete di base; (b), (c) e (d) illustrano l'incremento isolato rispettivamente di larghezza, profondità e risoluzione. (e) mostra il <i>compound scaling</i> , che scala simultaneamente tutte e tre le dimensioni seguendo un rapporto fisso [12].	46

4.14	Modello EfficientNetB3 trainato.	47
4.15	Modello EfficientNetB5 trainato.	47
4.16	Matrici di confusione per i modelli VGG16 (sinistra) e VGG19 (destra).	48
4.17	Matrici di confusione per il modello ResNet50 con training standard (sinistra) e con addestramento prolungato (destra).	49
4.18	Matrici di confusione per i modelli EfficientNetB3 (sinistra) e EfficientNetB5 (destra).	49
4.19	Matrici di confusione per il modello InceptionV3.	49
4.20	Andamento della <i>loss</i> (sinistra) e dell' <i>accuracy</i> (destra) durante la seconda fase di training del modello VGG19 , in cui viene scongelata una parte della rete convoluzionale. È evidente la presenza di overfitting, con la <i>training accuracy</i> che continua a migliorare, mentre la <i>validation accuracy</i> si stabilizza attorno al 70%.	53
4.21	Matrici di confusione per i modelli VGG16 (sinistra) e VGG19 (destra).	54
4.22	Matrice di confusione per i modelli ResNet50 (sinistra) e InceptionV3 (destra).	55
4.23	Matrici di confusione per i modelli EfficientNetB3 (sinistra) e EfficientNetB5 (destra).	55
4.24	Confronto delle prestazioni tra modelli addestrati su immagini RGB e multispettrali in termini di <i>accuracy</i> e <i>F1-score</i> separato per le classi “malato” e “sano”. Si osserva una maggiore stabilità nelle predizioni per i modelli multispettrali, con valori più bilanciati tra le due classi.	57

Capitolo 1

Introduzione

L'agricoltura di precisione rappresenta un settore in continua evoluzione, con un crescente interesse nell'adozione di tecnologie avanzate per il monitoraggio dello stato di salute delle colture. Tra queste, l'uso di immagini acquisite da droni equipaggiati con camere multispettrali si sta affermando come una delle soluzioni più promettenti per l'analisi non invasiva delle coltivazioni.

Negli ultimi anni, numerosi studi hanno dimostrato come le immagini da remoto, combinate con tecniche di intelligenza artificiale e apprendimento automatico, possano essere impiegate per identificare precocemente sintomi di malattie nelle piante, migliorando così l'efficacia degli interventi agronomici e riducendo l'uso di fitofarmaci.

Questa tesi si colloca all'interno di questo ambito di ricerca e si propone di studiare metodi automatici per la segmentazione e classificazione dello stato di salute di alberi da frutto, con particolare riferimento ai noccioli, coltivati in aree del Piemonte rinomate per la produzione di nocciole. L'obiettivo è confrontare diversi approcci di segmentazione basati su indici vegetazionali e tecniche di visione artificiale, valutandone l'efficacia e le criticità.

A supporto dell'analisi, è stato utilizzato un dataset originale composto da oltre 3.000 immagini raccolte tramite drone in diversi periodi della stagione vegetativa, comprendenti sia bande RGB che multispettrali. Le immagini sono state segmentate per isolare le chiome degli alberi, successivamente suddivise in sotto-immagini e classificate tramite reti neurali convoluzionali (CNN), utilizzando strategie di transfer learning e tecniche di data augmentation per migliorare le performance dei modelli.

Il lavoro si propone quindi di rispondere a domande di ricerca fondamentali, come: Quali metodi di segmentazione risultano più efficaci in presenza di disturbi come l'erba o l'ombra? Quali combinazioni di bande spettrali e architetture CNN offrono i migliori risultati per il riconoscimento di piante malate?

Nel corso della trattazione, saranno presentati i risultati ottenuti, confrontati con i metodi presenti in letteratura, e discusse le possibili implicazioni applicative di tali tecniche in un contesto reale.

1.1 Goal

L'obiettivo principale di questa tesi è la progettazione e realizzazione di una pipeline completa per la classificazione automatica dello stato di salute degli alberi di nocciolo, basata su immagini aeree acquisite da droni UAV equipaggiati con sensori RGB e multispettrali.

Il lavoro si articola in più fasi: l'analisi delle immagini, la segmentazione della chioma, l'estrazione dell'albero di interesse, la suddivisione in sottoimmagini significative e l'addestramento di modelli di deep learning per la classificazione binaria (sano/malato). Un ulteriore obiettivo è quello di confrontare diverse architetture convoluzionali (CNN), analizzando l'impatto del tipo di immagine (RGB vs multispettrale) e delle strategie di preprocessing sulla qualità della classificazione.

Particolare attenzione è stata posta all'integrazione di tecniche di transfer learning, data augmentation e regularization per migliorare le performance in presenza di un dataset ridotto. La tesi si propone quindi di valutare quali combinazioni di architettura, tipo di immagine e strategie di training siano più efficaci per il problema specifico analizzato.

1.2 Struttura della tesi

La tesi è strutturata in nove capitoli.

Nel Capitolo 1 viene introdotto il contesto dell'agricoltura di precisione e il problema della classificazione dello stato fitosanitario tramite immagini UAV.

Il Capitolo 2 descrive i dataset utilizzati, inclusi i dettagli delle campagne di acquisizione, le caratteristiche delle immagini (RGB e multispettrali) e le modalità di annotazione.

Nel Capitolo 3 si presentano i metodi di segmentazione testati, con particolare attenzione all'approccio NDVI e al metodo "Citrus", nonché alla procedura di estrazione dell'albero centrale e alla suddivisione delle chiome.

Il Capitolo 4 affronta le architetture di deep learning utilizzate, le tecniche di training, il setup sperimentale e le modifiche introdotte per adattare i modelli al dominio agricolo. In seguito vengono discussi i risultati ottenuti, con un confronto tra modelli, bande e tecniche di regolarizzazione, evidenziando i punti di forza e i limiti delle varie configurazioni.

Infine, nel Capitolo 5 si traggono le conclusioni, si riassumono i risultati chiave e si delineano i possibili sviluppi futuri, tra cui l'espansione del dataset, il miglioramento della regolarizzazione e l'utilizzo di modelli più avanzati per immagini multispettrali.

Capitolo 2

Dataset

Il dataset utilizzato in questo lavoro è composto da immagini di alberi di nocciolo viste dall'alto, acquisite tramite drone DJI P4 Multispectral durante la stagione vegetativa del **2022**. Le immagini si riferiscono a circa **185 piante** coltivate in piantagioni situate nella provincia di **Cuneo**, in Piemonte.

2.1 Descrizione generale del dataset

Le immagini sono state raccolte in due campi differenti e il dataset può essere quindi suddiviso in due sottoinsiemi principali:

- Dataset di Carrù
- Dataset di Farigliano

2.1.1 Dataset di Carrù

Il dataset di Carrù comprende **127 alberi di nocciolo**, ciascuno dei quali è stato ripreso in **tre diverse giornate**:

- 30 maggio 2022
- 22 giugno 2022
- 15 luglio 2022

Le immagini sono state acquisite in un campo localizzato nel comune di **Carrù**, in provincia di Cuneo (Piemonte).



Figura 2.1: Veduta aerea del campo di noccioli a Carrù.



(a) 30 Maggio 2022



(b) 22 Giugno 2022



(c) 15 Luglio 2022

Figura 2.2: Esempi di immagini acquisite nel campo di Carrù in tre differenti giornate.

2.1.2 Dataset di Farigliano

Il dataset di Farigliano include **58 alberi di nocciolo**, anch'essi osservati in **tre giornate distinte**:

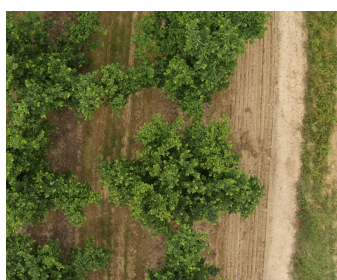
- 3 giugno 2022
- 24 giugno 2022

- 16 luglio 2022

Le immagini sono state scattate in un campo situato nel comune di **Farigliano**, in provincia di Cuneo.



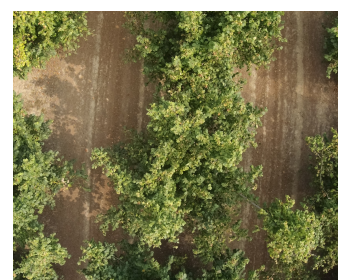
Figura 2.3: Veduta aerea del campo di noccioli a Farigliano.



(a) 3 giugno 2022



(b) 24 giugno 2022



(c) 16 luglio 2022

Figura 2.4: Esempi di immagini acquisite nel campo di Farigliano in tre differenti giornate.

2.2 Formato e tipologia delle immagini

Per ogni pianta osservata in ciascuna giornata di acquisizione, sono disponibili sei immagini distinte, ottenute mediante una fotocamera multispettrale montata su drone. Il dataset è pertanto strutturato per fornire una rappresentazione multispettrale completa di ciascun albero in ogni sessione di volo.

Le immagini disponibili per ogni campione sono le seguenti:

- **1 immagine RGB** in formato .JPG
- **5 immagini** in formato .TIF, ciascuna corrispondente a una banda specifica:
 - **Red Band (R)**: sensibile alla riflettanza della superficie nella banda del rosso visibile (600–690 nm).
 - **Green Band (G)**: corrisponde alla luce visibile verde (520– 600 nm), utile per il contrasto vegetazione/suolo.
 - **Blue Band (B)**: banda del blu visibile (450–520 nm), spesso usata per evidenziare corpi idrici o suoli nudi.
 - **Red-Edge Band (RE)**, centrata su lunghezze d’onda nell’intervallo tra il rosso e il vicino infrarosso (670–750 nm). Questa banda è particolarmente sensibile alle variazioni nella struttura interna delle foglie e risulta efficace nel rilevare stress precoci nella vegetazione.
 - **Near-Infrared Band (NIR)**, acquisita nella banda del vicino infrarosso (750–900 nm). Il NIR è fortemente riflesso dalla vegetazione sana a causa della struttura cellulare delle foglie ed è uno dei canali più utilizzati per calcolare indici di vegetazione come l’NDVI (Normalized Difference Vegetation Index).

Tutte le immagini multispettrali sono allineate spazialmente e presentano risoluzioni compatibili, consentendo l’analisi combinata delle diverse bande. La disponibilità separata delle bande consente inoltre di costruire indici vegetazionali o di sperimentare trasformazioni cromatiche avanzate per il miglioramento della segmentazione.

La figura 2.5 mostra un esempio delle sei immagini acquisite per uno stesso albero in una singola data, evidenziando le differenze visive tra le diverse bande.

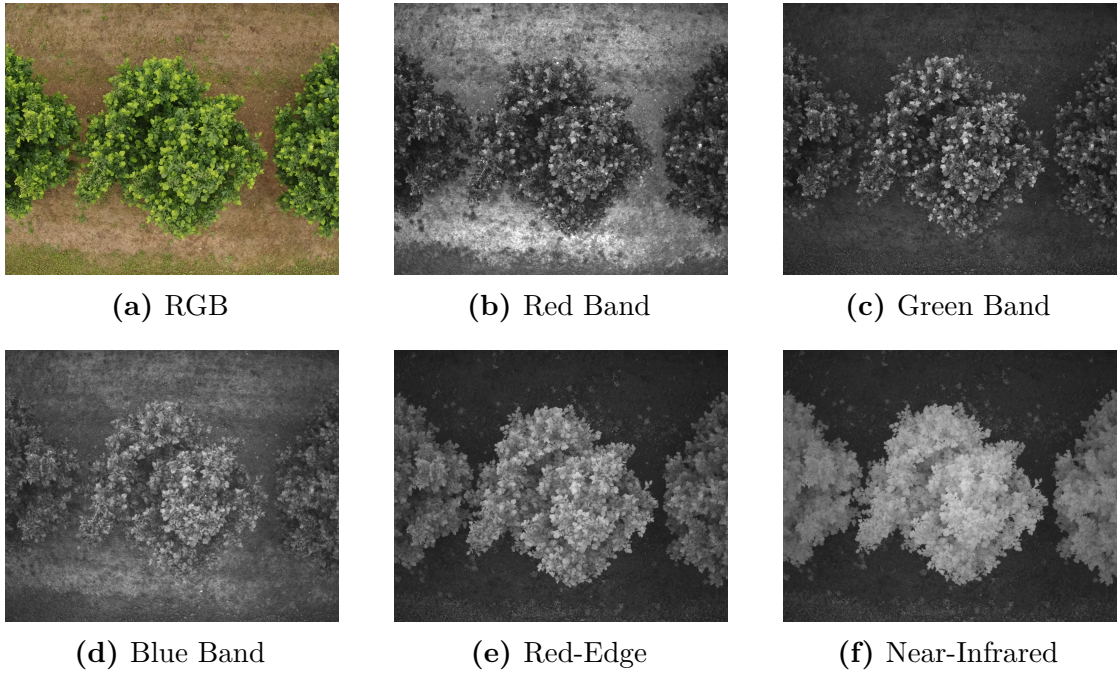


Figura 2.5: Esempio delle sei immagini acquisite per lo stesso albero

2.3 Esplorazione del Dataset

Al fine di valutare la qualità e l'affidabilità delle immagini disponibili, è stata effettuata un'analisi preliminare del dataset. Tale analisi ha evidenziato diverse problematiche.

Una delle principali criticità riscontrate riguarda la presenza di più alberi all'interno della stessa immagine. Nella maggior parte dei casi, infatti, le immagini non ritraggono una singola pianta, bensì una porzione di campo contenente più chiome (Figura 2.6).



Figura 2.6: Esempio di immagine contenente più alberi

Questa condizione rende spesso difficile distinguere i contorni tra le diverse piante, in particolare quando si verificano sovrapposizioni tra fogliame e rami (Figura 2.7). Per affrontare questa problematica, è stato necessario sviluppare un metodo specifico per identificare l'albero di riferimento da analizzare.



Figura 2.7: Esempio di immagine con alberi sovrapposti

Un'altra difficoltà riguarda le condizioni di illuminazione, che variano sensibilmente tra le diverse sessioni di acquisizione. Differenze di luminosità e angolazione

solare influenzano tutte le bande multispettrali (Figura 2.8).



Figura 2.8: La stessa pianta ripresa con differenti condizioni di illuminazione

Nell'immagine a sinistra, si osserva una scena ben illuminata e priva di ombre evidenti, mentre in quella a destra si nota chiaramente come la luce colpisca la pianta in modo non uniforme, generando zone d'ombra. Questo fenomeno può comportare una perdita di dettagli e, di conseguenza, una riduzione della qualità delle informazioni disponibili.

Un'ulteriore criticità emersa durante l'analisi del dataset riguarda la posizione non centrata degli alberi in alcune immagini. Come mostrato in Figura 2.9, capita frequentemente che l'albero di interesse non sia perfettamente centrato nel frame, ma si trovi parzialmente decentrato o addirittura in posizione marginale. Questa condizione può ridurre l'efficacia degli algoritmi di segmentazione automatica, oltre a rendere più complessa l'identificazione univoca dell'albero da considerare per la classificazione.



Figura 2.9: Esempio di immagine in cui l'albero non è centrato

Infine, è stato osservato che, per la maggior parte delle piante, le immagini relative alle diverse bande spettrali non risultano perfettamente sovrapponibili. Questo disallineamento può essere causato da diversi fattori: oltre al leggero ritardo tra le acquisizioni dovuto alla sequenzialità di cattura delle bande e al movimento del drone durante il volo, un ruolo significativo è giocato dalla diversa posizione fisica delle ottiche. Nei sensori multispettrali, infatti, ogni banda viene spesso acquisita da un obiettivo separato, montato in posizioni leggermente differenti (dell'ordine di alcuni centimetri), il che introduce parallasse e distorsioni prospettiche anche in presenza di condizioni di volo stabili.

Per evidenziare questo problema, si riporta in Figura 2.10 un confronto tra l'immagine RGB e la corrispondente acquisizione nella banda del vicino infrarosso (NIR). Nella parte superiore, le immagini RGB e NIR mostrano un rettangolo che individua un'area di interesse comune, definita utilizzando le stesse coordinate in pixel su entrambe le immagini. Nella parte inferiore, si mostra uno zoom sulla regione selezionata che rivela con maggiore chiarezza il disallineamento: le foglie dell'albero centrale nell'immagine NIR risultano visibilmente spostate verso destra di qualche centimetro rispetto alla corrispondente posizione nell'immagine RGB. Questo effetto, poco percepibile osservando le immagini intere, diventa evidente solo tramite l'ingrandimento, mettendo in luce la presenza di uno shift spaziale tra le acquisizioni.

Nonostante le immagini rappresentino la medesima scena, i contorni e le strutture delle piante appaiono spostati, segno che le acquisizioni non sono perfettamente allineate a livello di pixel. Tali artefatti, se non corretti tramite un'accurata registrazione delle immagini, possono compromettere l'accuratezza delle analisi

multispettrali e la qualità di indici vegetazionali che combinano più bande, come NDVI o GNDVI, soprattutto in corrispondenza dei bordi degli oggetti.

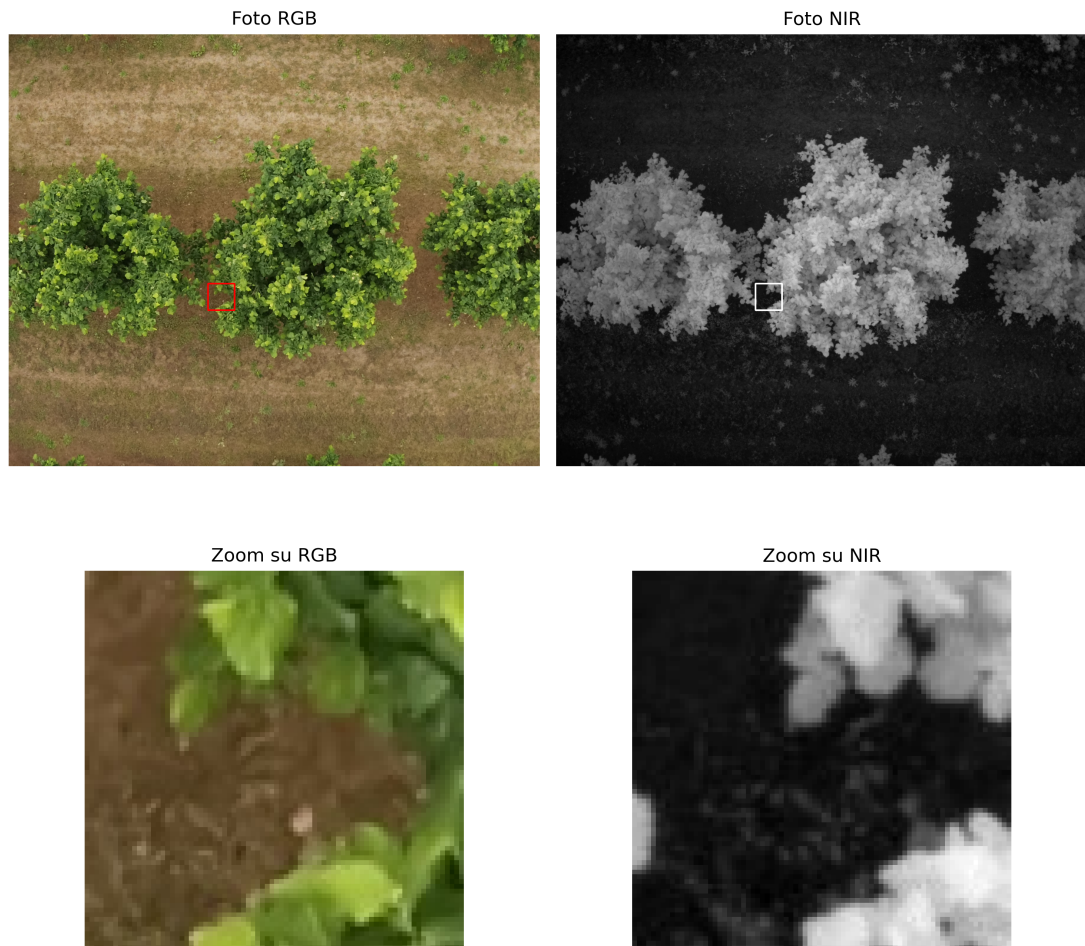


Figura 2.10: Evidenza visiva del disallineamento tra immagine RGB e immagine NIR. La parte inferiore mostra uno zoom sull'area evidenziata nella scena principale, mettendo in luce lo shift tra le due acquisizioni causato dalla diversa posizione delle ottiche.

Queste discrepanze costituiscono una sfida importante per le fasi successive di elaborazione, rendendo necessarie tecniche di correzione geometrica per garantire una coerenza spettrale tra bande.

Capitolo 3

Preprocessing

3.1 Metodi di segmentazione in letteratura

La segmentazione rappresenta un passaggio cruciale nell'ambito della Object-Based Image Analysis (OBIA), influenzando in modo determinante la qualità delle successive fasi di estrazione e classificazione delle caratteristiche.

Il tesista precedente aveva implementato un metodo di segmentazione basato sull'indice NDVI (Normalized Difference Vegetation Index), conservando i pixel con valori superiori a una soglia prefissata (0.2 o 0.3 a seconda del dataset). Questo approccio, descritto anche nel lavoro di Morisio et al. NDVIhazelnut, verrà approfondito in una sezione successiva.

Nel nostro studio, abbiamo inoltre esplorato e confrontato metodi alternativi tratti dalla letteratura.

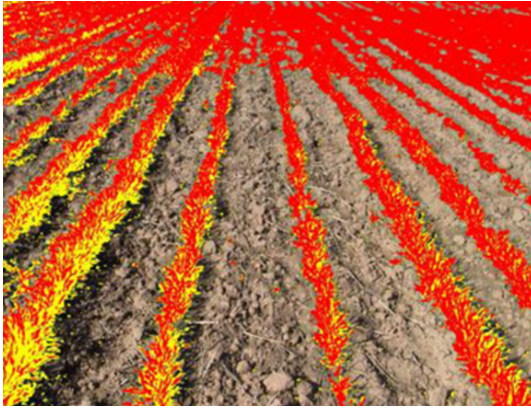
3.1.1 Segmentazione tramite combinazione di indici vegetazionali

Il paper *Automatic segmentation of relevant textures in agricultural images* [1] propone un metodo per identificare le principali texture presenti nelle immagini agricole (piante, suolo e cielo), attraverso un approccio basato su indici spettrali. Questo metodo, non progettato per alberi ma per piante erbacee, si articola in tre fasi principali:

1. **Identificazione delle piante:** vengono calcolati i seguenti indici spettrali: Excess Green (ExG), Excess Green minus Red (ExGR), CIVE e VEG. Gli indici sono poi combinati con pesi determinati tramite una misura di uniformità dell'istogramma, migliorando la segmentazione rispetto all'uso di un singolo indice.

2. **Separazione del suolo e del cielo:** si utilizzano gli indici Excess Red (ExR) e Excess Blue (ExB). Il rapporto tra i pixel rossi e blu permette di determinare la presenza del cielo nell'immagine.
3. **Clustering fuzzy:** viene applicato un algoritmo di clustering fuzzy per distinguere diverse tonalità di verde e caratteristiche del suolo, assegnando a ciascun pixel un grado di appartenenza a più classi.

Errore medio nella segmentazione delle aree verdi: 8.31%.



(a) Classificazione in due diverse classi di piante verdi.



(b) Classificazione in due diverse classi per il terreno.

Figura 3.1: Esempi di segmentazione e classificazione

3.1.2 Metodo per ulivi e rilevamento di *Xylella fastidiosa*

Il paper *A New Processing Method to Segment Olive Trees and Detect Xylella Fastidiosa in UAVs Multispectral Images* [2] propone un metodo per la segmentazione degli ulivi e la rilevazione dei sintomi della *Xylella fastidiosa*, basato sull'utilizzo di immagini multispettrali acquisite tramite droni. Di seguito vengono descritti i principali step del processo:

1. **Allineamento:** il paper evidenzia che la mancata corrispondenza tra le bande spettrali è uno dei problemi principali nelle immagini multispettrali acquisite da camere UAV, dovuto a leggere differenze geometriche tra i sensori. L'allineamento dei pixel tra bande risulta quindi un passaggio obbligato per garantire l'affidabilità delle analisi successive. Anche nel nostro lavoro, questa fase è stata considerata preliminare e necessaria prima di applicare qualsiasi tecnica di segmentazione o classificazione.
2. **Modello 3D:** generazione di un modello tridimensionale del sito, con classificazione dei punti in base all'altezza per isolare le chiome.

3. **Segmentazione:** la fase di segmentazione degli alberi è articolata in cinque sotto-step distinti:
 - **Classificazione per altezza:** i punti del modello 3D vengono classificati in base all'elevazione, confrontandola con una soglia appositamente definita per separare le chiome da altri oggetti.
 - **Oversegmentazione con algoritmo di Felzenszwalb:** applicata alla banda NIR per ottenere una segmentazione preliminare basata sulle discontinuità locali.
 - **Segmentazione guidata da elevazione, NIR e NDVI:** si esegue una segmentazione basata congiuntamente sui valori di elevazione, NIR e NDVI, ottenendo una prima classificazione dei segmenti potenzialmente appartenenti agli alberi.
 - **Classificazione LDA dei segmenti di Felzenszwalb:** i segmenti ottenuti al punto (ii) vengono classificati tramite la Linear Discriminant Analysis(LDA).
 - **Identificazione finale degli alberi:** solo i segmenti che risultano positivi sia nel terzo che nel quarto punto vengono considerati "Albero".
4. **Valutazione dello stato di salute:** dopo la segmentazione, un classificatore LDA viene applicato allo stack multispettrale, addestrato su pixel di alberi sani e malati (OQDS). Il risultato è una mappa di probabilità a livello di pixel; ogni regione segmentata viene infine classificata come "sana" o "malata" in base alla media delle probabilità e a una soglia prestabilita.



Figura 3.2: Pipeline di segmentazione e classificazione dello stato di salute per ulivi.

3.1.3 Metodo "Citrus" per la segmentazione di alberi di agrumi

Il paper *Citrus Tree Segmentation from UAV Images Based on Monocular Machine Vision in a Natural Orchard Environment* [3] propone un metodo per la segmentazione automatica di alberi di agrumi a partire da immagini RGB acquisite da UAV in condizioni ambientali reali, caratterizzate da forte variabilità di illuminazione e presenza di erbe infestanti. Il metodo si articola in tre fasi principali: pre-processing, estrazione delle regioni di interesse (RoI) e segmentazione supervisionata.

1. **Pre-processing con SRIHE:** per aumentare il contrasto tra le chiome degli alberi e il suolo senza alterare la componente cromatica, viene applicata la *Selective Region Intensity Histogram Equalization* (SRIHE). L'immagine RGB viene convertita nello spazio HSI (Hue, Saturation, Intensity); si selezionano i pixel con tonalità H compresa tra $\pi/2$ e π (che corrisponde alle tonalità di verde, tipiche della chioma), e solo in questa regione si applica una equalizzazione dell'intensità. Questo consente di migliorare la visibilità delle chiome sotto differenti condizioni di luce, riducendo l'impatto di ombre e riflessi. L'immagine viene infine riconvertita in RGB per i successivi step.
2. **Estrazione delle regioni di interesse (RoI):** si utilizza il metodo *ER-GCM* (Extraction based on Relative Green Chromaticity Map), che sfrutta la differenza cromatica relativa tra il canale verde e quello rosso per isolare le chiome. Tuttavia, tale approccio è sensibile alla presenza di erbe infestanti. Per mitigare questo problema, quando la separazione risulta insufficiente, viene applicato un miglioramento basato su due tecniche:
 - *Retinex multi-scala*, che corregge la variabilità luminosa e migliora la percezione dei dettagli sotto ombre o luce diretta;
 - *Mappatura cromatica adattiva*, che accentua il contrasto tra chiome e fondo erboso, aiutando la corretta estrazione delle RoI.
3. **Segmentazione supervisionata con SVM:** sulle regioni di interesse estratte vengono calcolate 78 caratteristiche in totale:
 - **14 caratteristiche di colore**, incluse medie e varianze dei canali R, G, B, H, S, a^* e b^* (da spazi RGB, HSI e Lab);
 - **64 caratteristiche di texture**, derivate da due fonti:
 - 5 statistiche dalla *Gray-Level Co-occurrence Matrix* (GLCM): contrasto, correlazione, energia, omogeneità e entropia;
 - 59 caratteristiche da *Local Binary Pattern* (LBP) uniformi, sensibili alla distribuzione locale di intensità.

Queste feature vengono utilizzate per addestrare un classificatore *Support Vector Machine* (SVM), che assegna a ciascuna RoI un'etichetta binaria: "albero" o "non albero".

Prestazioni: Il metodo ha mostrato buoni risultati di accuratezza e generalizzazione su immagini acquisite in condizioni naturali, riportando:

- **Correct Segmentation Rate:** $85.27\% \pm 9.43$
- **Intersection over Union (IoU):** 81.76 ± 11.06

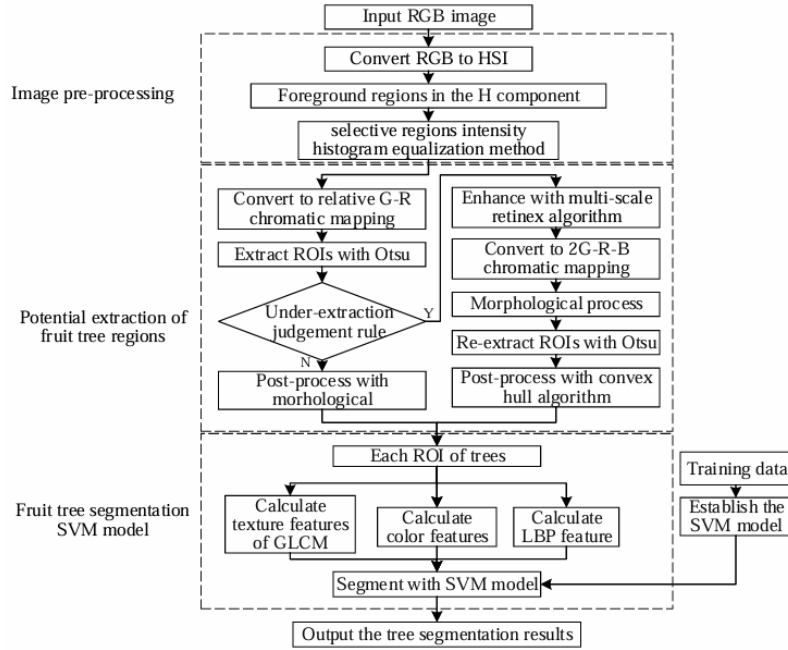


Figura 3.3: Schema del metodo proposto

Questo approccio, pur operando su immagini RGB (senza multispettrale), si dimostra robusto grazie all'uso di tecniche di pre-elaborazione mirate e di una fase di classificazione supervisionata basata su un ampio set di feature. L'uso della SRIHE è particolarmente interessante per contesti agricoli reali, dove la variabilità di luce può compromettere l'efficacia di metodi più semplici. Il metodo rappresenta quindi una buona base da cui partire per applicazioni analoghe in contesti non multispettrali.

3.2 Implementazione dei metodi di segmentazione

A seguito dell'analisi comparativa dei metodi di segmentazione presenti in letteratura, si è scelto di procedere con l'implementazione di due approcci distinti: un metodo basato su indice NDVI, già sperimentato nel lavoro precedente, e il metodo "Citrus", descritto nel dettaglio nella sezione precedente. Queste due strategie presentano caratteristiche complementari: il primo sfrutta la semplicità e l'efficacia degli indici vegetazionali su immagini multispettrali, mentre il secondo propone un approccio più articolato, adatto a immagini RGB in contesti naturali complessi.

In questa sezione vengono descritte nel dettaglio le scelte implementative, le librerie utilizzate, le modalità di pre-processing e i criteri adottati per la segmentazione e valutazione dei risultati. L'obiettivo è confrontare l'efficacia dei due metodi in scenari reali, tenendo conto delle specificità delle immagini a disposizione (multispettrali vs RGB), delle condizioni ambientali e delle caratteristiche delle colture analizzate.

In questa fase del lavoro, ci si concentrerà esclusivamente sulla valutazione della segmentazione in quanto tale, ossia sulla capacità dei metodi implementati di isolare correttamente le porzioni d'immagine corrispondenti alle chiome degli alberi. Non verranno ancora considerate operazioni successive come l'estrazione del singolo albero centrale o la classificazione dello stato di salute: queste verranno affrontate in fasi successive del pipeline.

Nel seguito, verranno presentati in ordine:

- il metodo di segmentazione basato su soglia NDVI;
- il metodo "Citrus" per immagini RGB.

3.2.1 Segmentazione mediante indice NDVI

L'indice di vegetazione NDVI (Normalized Difference Vegetation Index) rappresenta una delle tecniche più consolidate per la discriminazione della vegetazione in immagini multispettrali. È calcolato come:

$$\text{NDVI} = \frac{\text{NIR} - \text{Red}}{\text{NIR} + \text{Red}} \quad (3.1)$$

dove *NIR* indica il valore del pixel nella banda del vicino infrarosso e *Red* quello nella banda del rosso visibile. Valori elevati di NDVI indicano vegetazione densa e attiva fotosinteticamente, mentre valori prossimi allo zero o negativi sono tipici di suolo nudo, ombre o superfici artificiali.

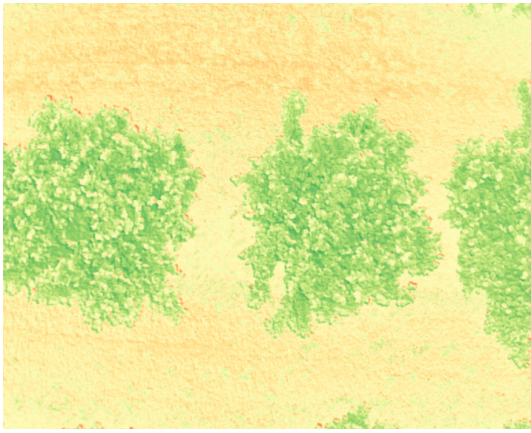
Nel nostro caso, l'NDVI è stato calcolato pixel per pixel a partire da immagini multispettrali acquisite tramite drone. L'implementazione è stata effettuata in ambiente Python utilizzando le librerie `NumPy`, `rasterio` e `OpenCV`.

Prima del calcolo dell'indice NDVI, è stato necessario affrontare un problema critico legato alle immagini multispettrali: il disallineamento tra bande. Questo fenomeno, dovuto a leggere differenze ottiche o a movimenti del drone in fase di acquisizione, può compromettere la correttezza del calcolo pixel-wise dell'NDVI, generando artefatti visivi e riducendo la qualità della segmentazione. Per risolvere questo problema, è stato implementato un sistema automatico di allineamento geometrico tra le bande multispettrali, basato sull'estrazione e corrispondenza di keypoint tramite l'algoritmo SIFT (Scale-Invariant Feature Transform) e successiva omografia per la trasformazione dell'immagine. In particolare, le bande da allineare

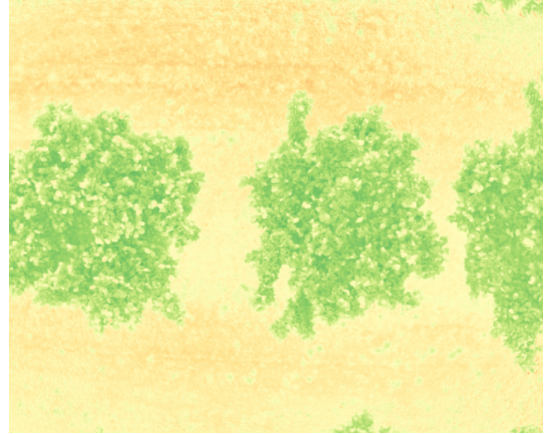
(ad esempio NIR) sono registrate rispetto alla componente RGB di riferimento. Questo pre-processing ha consentito di ottenere una sovrapposizione più precisa delle informazioni spettrali, migliorando la coerenza spaziale del calcolo NDVI e la qualità delle maschere ottenute.

Per meglio visualizzare l'impatto dell'allineamento geometrico tra bande, si è scelto di confrontare due versioni del GNDVI, calcolate rispettivamente con e senza pre-processing. Rispetto all'NDVI, il GNDVI risulta particolarmente efficace nel mettere in evidenza gli artefatti causati dal disallineamento, grazie alla maggiore sensibilità della banda verde a variazioni spaziali e radiometriche. Il GNDVI (*Green Normalized Difference Vegetation Index*) è una variante dell'NDVI in cui, al posto della banda rossa, viene utilizzata la banda verde; la formula diventa quindi:

$$\text{GNDVI} = \frac{\text{NIR} - \text{Green}}{\text{NIR} + \text{Green}}$$



(a) GNDVI senza allineamento.



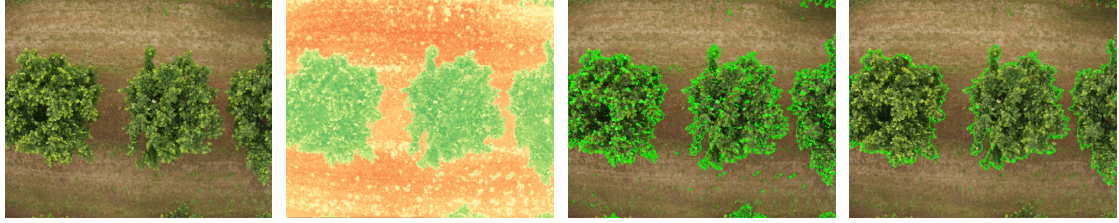
(b) GNDVI con allineamento.

Figura 3.4: Confronto tra GNDVI calcolato con e senza allineamento delle bande. Il disallineamento tra la banda verde e la NIR genera artefatti visibili nei contorni degli alberi. L'allineamento migliora la qualità della segmentazione.

Soglia e binarizzazione

Per isolare la vegetazione, è stata applicata una soglia empirica al valore di NDVI, convertendo l'immagine in una maschera binaria. Sono stati testati diversi valori soglia; i migliori sono stati 0.2 e 0.3: valori inferiori possono includere rumore di fondo, mentre soglie più alte riducono l'effetto di ombre e luce disomogenea, ma rischiano di eliminare porzioni reali della chioma. Nei casi più problematici, soglie superiori (fino a 0.4) sono state utilizzate per migliorare la separabilità delle chiome.

Successivamente, sono state applicate operazioni morfologiche (rimozione dei piccoli oggetti, chiusura) per migliorare la coerenza spaziale della maschera ottenuta.



(a) RGB originale (b) NDVI calcolato (c) Contorni estratti (d) Contorni filtrati

Figura 3.5: Workflow del metodo: partendo dall'immagine RGB, si calcola la mappa NDVI, si estraggono i contorni delle aree vegetate mediante threshold e si filtrano per ottenere solo quelli rilevanti.

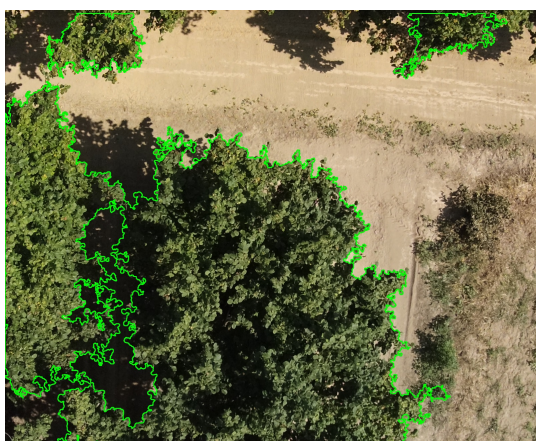
Limitazioni e scenari tipici

Il metodo basato su NDVI presenta il vantaggio di essere semplice, rapido e adatto all'elaborazione massiva di immagini multispettrali. Tuttavia, mostra alcune limitazioni, soprattutto in contesti complessi. In particolare, l'utilizzo di soglie fisse rende il metodo sensibile alle condizioni di acquisizione, come variazioni di luce o presenza di ombre. L'adozione di soglie più conservative consente di ridurre la contaminazione da oggetti non vegetali, ma al costo di una possibile perdita informativa ai bordi della chioma. Questa scelta rappresenta un compromesso accettabile per gli scopi della nostra analisi, in cui l'accurata localizzazione della massa vegetale prevale sulla completezza pixel-wise.

Le difficoltà nella separazione dell'albero di interesse rispetto ad altri elementi vegetali presenti nell'immagine si presentano sia in questo metodo sia nel metodo "Citrus". Di seguito si riportano tre scenari tipici che illustrano le principali problematiche comuni a entrambi gli approcci:

- **Scenario 1 – Albero isolato:** se l'albero è l'unica vegetazione presente, il soggetto è facilmente identificabile e segmentabile con precisione;
- **Scenario 2 – Alberi separati:** se sono presenti più alberi ben distinti, è possibile stimare i bounding box dell'albero centrale e isolarlo efficacemente;
- **Scenario 3 – Alberi intrecciati:** in presenza di chiome sovrapposte o contigue, diventa molto difficile, se non impossibile, distinguere automaticamente l'albero di interesse con metodi puramente automatici.

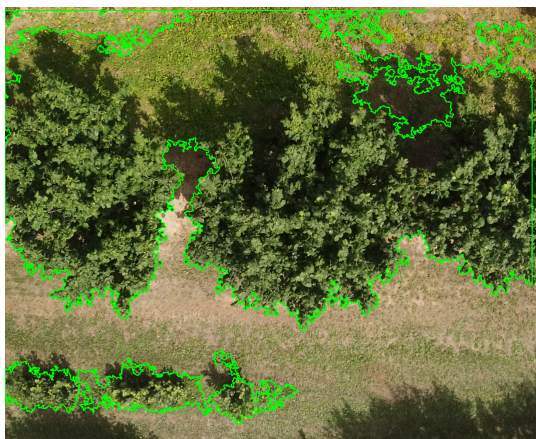
Per concludere, nella figura seguente sono riportati alcuni esempi rappresentativi dei risultati ottenuti con il metodo NDVI.



(a) Buona segmentazione anche in presenza di ombre. Gli alberi risultano ben separati.



(b) Segmentazione generalmente corretta, ma alcuni rami secchi non vengono rilevati (evidenziati in rosso).



(c) Soglia troppo bassa (0.2): porzioni di erba vengono erroneamente incluse.



(d) Segmentazione migliorata con soglia 0.4. In alcuni casi è necessario alzare la soglia rispetto ai valori usuali (0.2–0.3).

Figura 3.6: Esempi di segmentazione NDVI in diversi scenari. I primi due mostrano risultati positivi, con alcune eccezioni nei dettagli. Gli ultimi due evidenziano l'importanza della soglia scelta per un corretto bilanciamento tra accuratezza e inclusività.

3.2.2 Segmentazione mediante il metodo *Citrus*

Per confrontare approcci più complessi rispetto all'NDVI, è stato implementato un metodo ispirato all'articolo scientifico analizzato nella sezione precedente. L'approccio *Citrus*, proposto nel lavoro originale [3], presenta un workflow articolato

per la segmentazione degli alberi di agrumi da immagini UAV, come illustrato in Figura 3.7.

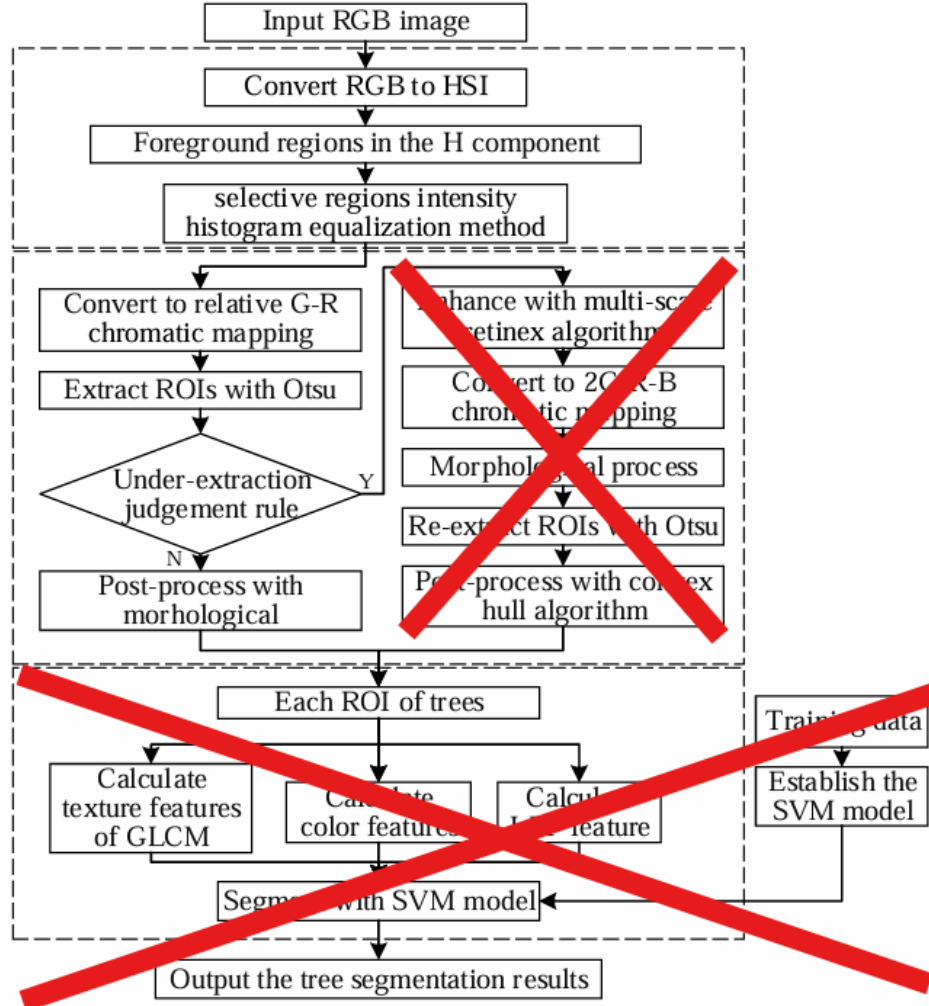


Figura 3.7: Schema del workflow completo del metodo Citrus. Le componenti contrassegnate con una X rossa indicano le parti non implementate nel presente lavoro.

Il metodo è stato parzialmente implementato, il primo passo dell'implementazione riguarda il **preprocessing** dell'immagine acquisita. In questa fase, l'immagine originale in formato RGB viene convertita nello spazio colore HSI (Hue, Saturation, Intensity), che consente di separare le informazioni cromatiche (tonalità e saturazione) da quelle di intensità luminosa. In particolare, si utilizza la componente *Hue* per identificare le regioni di primo piano, ovvero le aree probabilmente corrispondenti alla vegetazione, sfruttando la caratteristica del colore verde degli

alberi. Successivamente, si applica un'*equalizzazione selettiva dell'istogramma* sulla componente di intensità, una tecnica che migliora il contrasto locale dell'immagine distribuendo in modo più uniforme i livelli di luminosità. Questo processo permette di evidenziare meglio i dettagli nelle zone con basso contrasto, facilitando le fasi successive di segmentazione.

La seconda fase, corrispondente al percorso sinistro del metodo Citrus, si concentra sull'estrazione delle regioni di interesse (ROI) attraverso una serie di passaggi mirati. Innanzitutto, l'immagine viene trasformata in una mappatura cromatica relativa basata sulla differenza tra i canali verde e rosso (G-R), che accentua ulteriormente la presenza di vegetazione rispetto allo sfondo. Su questa immagine trasformata si applica una sogliatura adattiva tramite l'algoritmo di Otsu, che determina automaticamente la soglia ottimale per separare le regioni vegetative dal resto dell'immagine. A seguito di questa segmentazione iniziale, si impiegano delle regole di giudizio basate su criteri geometrici e di forma per eliminare falsi positivi o regioni spurie, migliorando così la qualità delle ROI estratte. Infine, si esegue un post-processing morfologico, mediante operazioni di apertura e chiusura, per rifinire i contorni delle regioni segmentate e rimuovere eventuali rumori residui, ottenendo così una segmentazione più coerente e pulita.

Due componenti principali del workflow originale non sono state implementate:

- **Percorso di enhancement avanzato (ramo destro):**

- Algoritmo multi-scale retinex per enhancement
- Conversione in mappatura cromatica 2G-R-B
- Processi morfologici avanzati con convex hull

- **Classificazione finale:**

- Estrazione features multimodali (GLCM, colore, forma)
- Addestramento classificatore SVM
- Classificazione supervisionata dei segmenti

La scelta di non implementare il percorso destro è stata dettata dalla volontà di adattare il workflow alle specifiche caratteristiche del dataset utilizzato, privilegiando soluzioni più semplici e robuste che meglio si adattano a immagini ravvicinate. La fase di classificazione SVM richiederebbe un dataset annotato manualmente per l'addestramento, non disponibile nel contesto di questo lavoro; per questo non è stato realizzato.

L'obiettivo di questa implementazione è valutare l'efficacia delle prime due fasi nel generare una segmentazione coerente degli alberi, confrontandola con i risultati ottenuti dal metodo NDVI basato su soglia.

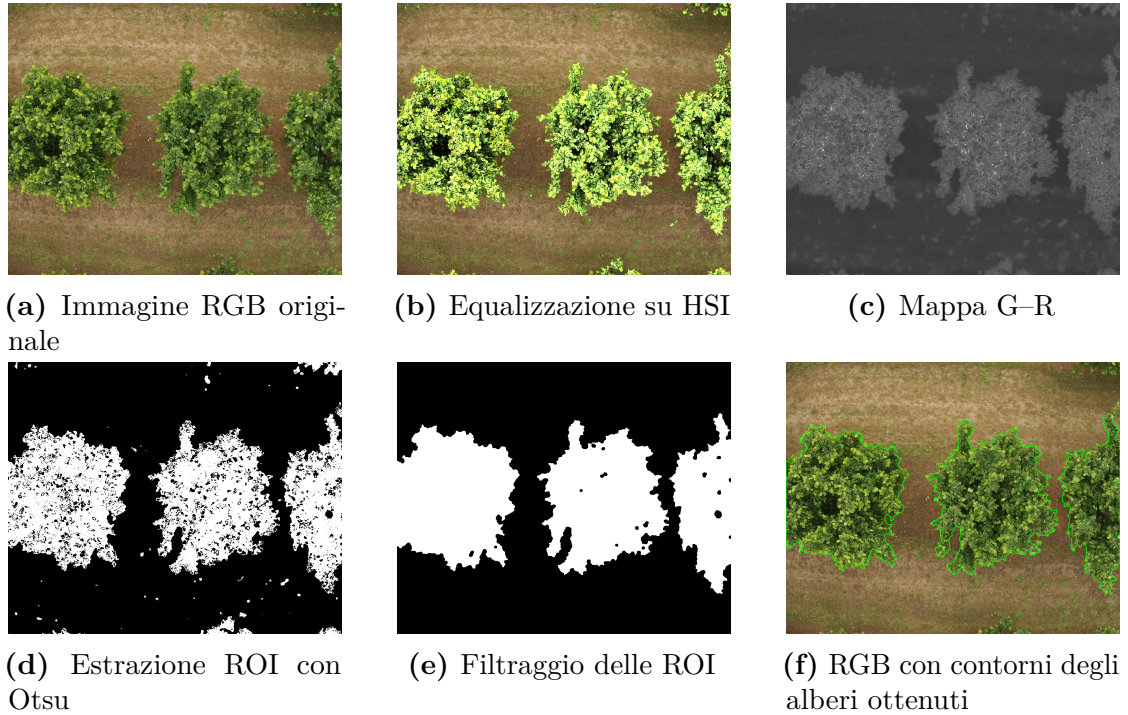


Figura 3.8: Esempio completo del workflow applicato a una singola immagine RGB. Le fasi mostrate comprendono: (a) immagine RGB originale, (b) conversione in spazio HSI ed equalizzazione selettiva della componente di intensità, (c) generazione della mappa cromatica G-R, (d) sogliatura con algoritmo di Otsu, (e) filtraggio morfologico delle ROI, (f) visualizzazione dei contorni degli alberi sovrapposti all'immagine RGB.

Conclusioni sul metodo *Citrus*

Il metodo *Citrus* si è dimostrato efficace in diversi casi, riuscendo a delineare in modo più preciso i contorni degli alberi rispetto al metodo basato sull'NDVI, specialmente in presenza di immagini RGB ben bilanciate e prive di forti disomogeneità luminose. La semplicità del workflow, unita alla possibilità di operare anche in assenza di bande multispettrali, lo rende una valida alternativa in contesti operativi specifici.

Tuttavia, il metodo presenta una criticità significativa: in presenza di alberi colpiti da luce solare intensa e disomogenea, la componente cromatica può risultare alterata, portando a una perdita rilevante di informazione nelle regioni interessate. Questo comporta errori di segmentazione anche gravi, con intere chiome che possono risultare escluse o frammentate.

Per tale motivo, a partire dalle fasi successive di questo lavoro, si è deciso di adottare stabilmente il metodo basato su NDVI come approccio principale per la

segmentazione degli alberi, garantendo maggiore robustezza in condizioni ambientali variabili.



(a) Segmentazione riuscita: contorni ben delineati.



(b) Chiome individuate correttamente grazie a condizioni di luce ottimali.



(c) Albero centrale non ben separato: rispetto al NDVI risulta meno definito.



(d) Errore grave: l'erba in ombra viene segmentata come chioma.

Figura 3.9: Confronto qualitativo dei risultati ottenuti con il metodo Citrus: due esempi riusciti (in alto) e due falliti (in basso).

3.3 Estrazione automatica dell'albero centrale

Per ogni immagine segmentata, è necessario identificare l'albero da considerare per l'analisi. A tal fine, sono stati esplorati due approcci distinti.

Metodo 1 – Estrazione basata sulla posizione centrale e contorni

Il primo metodo si basa sull'assunzione che l'albero di interesse si trovi approssimativamente al centro dell'immagine, ipotesi coerente con la modalità di acquisizione del dataset. Il processo inizia selezionando, tra tutti i contorni estratti nella fase di segmentazione, quello più prossimo al centro geometrico dell'immagine.

Una volta individuato questo contorno, si applicano due *sliding window* (SW) verticali, una verso sinistra e una verso destra rispetto al centro. Per ciascuna finestra si conta il numero totale di pixel che appartengono al contorno selezionato. Si ottiene così un profilo di densità lungo l'asse verticale.

Vengono poi individuati i primi minimi locali partendo dal centro verso ciascun lato, che corrispondono ai punti in cui la densità di pixel appartenenti al contorno diminuisce significativamente. Questi punti vengono utilizzati come limiti sinistro e destro del rettangolo che racchiude l'albero:

$$\text{left_bound} = \text{index}(\min(\text{SW}_{\text{left}})) \quad \text{right_bound} = \text{index}(\min(\text{SW}_{\text{right}})) \quad (3.2)$$

Lo stesso procedimento viene replicato lungo l'asse orizzontale, per stimare i confini superiore e inferiore dell'albero.

In casi rari, può accadere che il contorno dell'albero non sia centrato perfettamente nell'immagine, causando una mancata individuazione nella fase iniziale. Per gestire questa eventualità, è stato introdotto un parametro di spostamento (*shifting starting point*) che consente di traslare il punto di partenza delle finestre sia lungo l'asse orizzontale che verticale, fino a individuare una regione significativa del contorno.

Questo metodo si è dimostrato efficace nella maggior parte dei casi, grazie alla regolarità della disposizione degli alberi e alla coerenza della posizione centrale nei dati acquisiti.

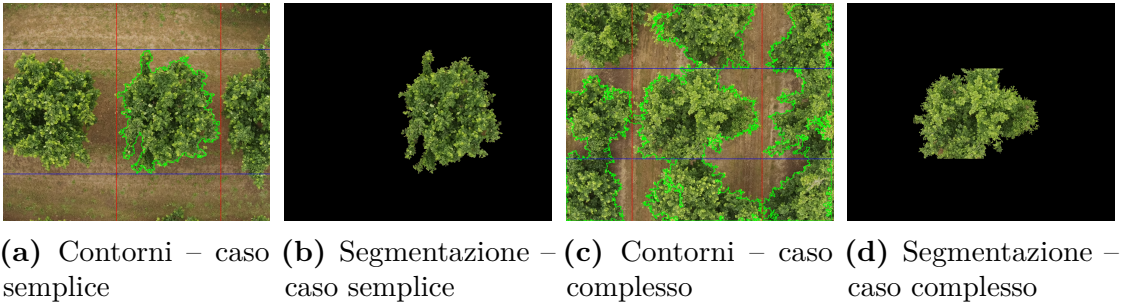


Figura 3.10: Esempi di estrazione automatica dell'albero centrale tramite sliding window. A sinistra un caso in cui l'albero è ben separato e i contorni risultano chiari. A destra un caso più complesso in cui gli alberi risultano sovrapposti.

Metodo 2 – Geolocalizzazione e coordinate note

Il secondo approccio sfrutta l'informazione geografica. Tutti gli alberi presenti nell'area di interesse sono stati preliminarmente mappati utilizzando Google Earth [4], dove sono state annotate le coordinate geografiche dei centri delle chiome. Le annotazioni sono state ottenute mediante un metodo semi-automatico: per ogni fila di alberi, sono stati identificati manualmente il primo e l'ultimo albero. Considerando che gli alberi sono disposti in linea retta e a una distanza regolare di circa 4 metri l'uno dall'altro, le posizioni intermedie sono state calcolate automaticamente tramite interpolazione lineare. Poiché le immagini acquisite dal drone contengono i metadati GPS del punto centrale della foto, è possibile identificare quale albero si trova più vicino al centro della foto.



(a) Porzione del campo di Carrù



(b) Porzione del campo di Farigliano

Figura 3.11: Mappatura dei campi mediante metodo semi-automatico GPS + interpolazione.

Conoscendo, inoltre, alcune informazioni tecniche del drone [5]:

- l'orientamento dei punti cardinali nell'immagine,
- e la trasformazione da pixel a centimetri fornita dal costruttore del drone, definita come:

$$\text{cm/pixel} = \frac{H}{18,90} \quad (3.3)$$

dove H è l'altezza di volo del drone rispetto al suolo, informazione disponibile nei metadati dei file TIF.

è possibile convertire le coordinate geografiche in coordinate locali sull'immagine, collocando così il centro dell'albero di interesse. Intorno a tale punto viene costruita una finestra quadrata di 4×4 metri per delimitare l'albero.



Figura 3.12: Esempio illustrativo relativo al secondo metodo di estrazione automatica dell'albero centrale.

Confronto e scelta del metodo

Il secondo metodo, pur fornendo una stima teoricamente più accurata, ha mostrato alcune imprecisioni nella pratica, soprattutto in presenza di errori GPS o variazioni nella disposizione effettiva degli alberi. Per questo motivo, **è stato preferito l'utilizzo del primo metodo** nella maggior parte dei casi, mentre il secondo è stato impiegato principalmente nelle situazioni in cui le chiome risultavano sovrapposte o troppo vicine per essere correttamente separate dal solo centro immagine.

3.4 Image Subdivision

Completata la fase di segmentazione e isolamento dell'albero centrale, l'ultimo passo del processo consiste nella suddivisione dell'albero segmentato in porzioni più piccole, da utilizzare successivamente per la classificazione dello stato di salute. Questo procedimento prende il nome di *Grid Subdivision*.

Nei lavori precedenti, ogni albero veniva suddiviso in un numero variabile di parti, in base alla dimensione della chioma. Per garantire la coerenza con le etichette già disponibili, abbiamo mantenuto lo stesso schema di suddivisione adottato in precedenza.

Il metodo prevede la suddivisione delle immagini segmentate in una griglia di dimensione $N \times N$, in modo che ciascun riquadro contenga una parte significativa della pianta. Questo approccio consente di localizzare in quali porzioni dell'albero si manifestano segni di malattia, permettendo l'assegnazione di etichette specifiche a ciascun riquadro.

Criteri di Suddivisione

Il primo passo consiste nel determinare il valore ottimale di N , sulla base dell'area (in pixel) occupata dalla chioma segmentata. A tal fine sono state definite due soglie:

- **Soglia superiore:** 300.000 pixel
- **Soglia inferiore:** 100.000 pixel

La regola seguita per scegliere N è la seguente:

- Se l'area della chioma supera la soglia superiore, si imposta $N = 3$, ottenendo una griglia 3×3 (9 riquadri);
- Se l'area è compresa tra la soglia inferiore e quella superiore, si imposta $N = 2$, con una griglia 2×2 (4 riquadri);
- Se l'area è inferiore alla soglia inferiore, si imposta $N = 1$, mantenendo l'intera immagine come unico riquadro.

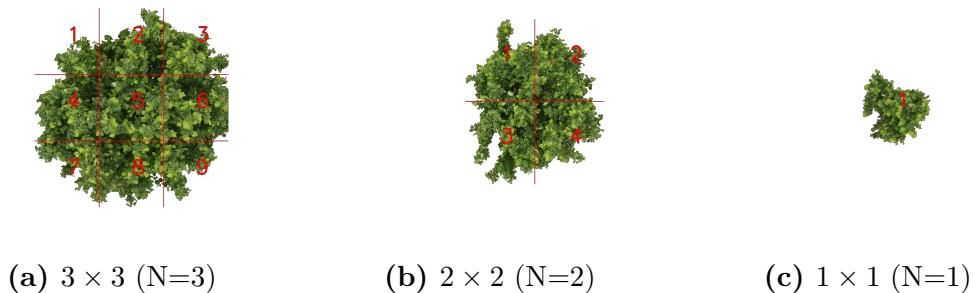


Figura 3.13: Esempi di suddivisione dell'immagine segmentata in griglie $N \times N$, in funzione dell'area occupata dalla chioma. Si noti come il valore di N cresca con l'aumentare della dimensione dell'albero.

Filtraggio delle Immagini Incoerenti

Durante la fase di suddivisione delle immagini, è emersa una criticità legata alla coerenza tra il numero di sottoimmagini generate e il numero di etichette disponibili nel file Excel, nel quale erano state annotate manualmente le informazioni sullo stato fitosanitario delle piante.

Tali etichette provengono da un lavoro precedente, realizzato con il supporto di esperti botanici, e rappresentano una risorsa preziosa non facilmente replicabile. In

particolare, per ciascuna immagine segmentata, l'Excel riporta un numero fisso di etichette che riflette il metodo di suddivisione usato all'epoca (ad esempio 2×2). Tuttavia, la nostra suddivisione automatica, basata sulla dimensione della chioma, può generare un numero diverso di sottoimmagini (ad esempio 3×3), creando un disallineamento non gestibile tra immagini e relative etichette.

Poiché non era possibile ripetere l'intero processo di etichettatura coinvolgendo nuovamente i botanici per valutare ogni singola sottoimmagine, si è optato per l'eliminazione di tutte le immagini affette da questa incongruenza. Di conseguenza, a partire da un totale iniziale di 4113 immagini suddivise, ne sono state mantenute 3445, con una perdita complessiva di circa il 16%.

Capitolo 4

Training dei Modelli e Strategie di Deep Learning

4.1 Introduzione al Deep Learning

Il *Deep Learning* è una sottoarea del machine learning che si basa sull'utilizzo di modelli chiamati **reti neurali artificiali**, composti da molteplici strati di neuroni interconnessi. Questi modelli si ispirano, in modo molto semplificato, alla struttura del cervello umano: ciascun neurone riceve input in forma numerica, li elabora tramite una funzione di attivazione non lineare e trasmette il risultato ai neuroni successivi.

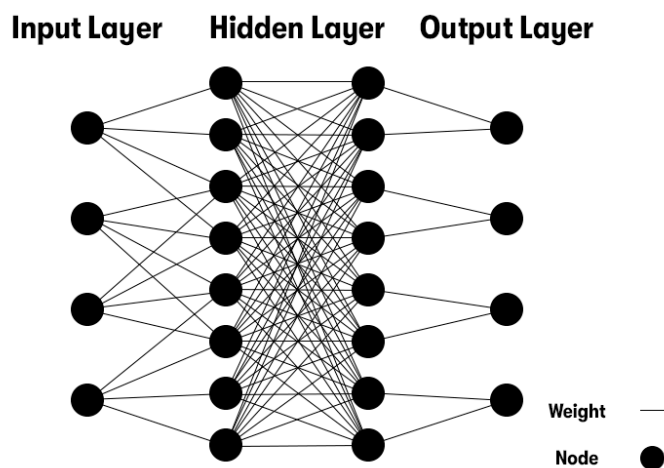


Figura 4.1: Schema concettuale di una rete neurale [6]

A differenza degli approcci tradizionali di machine learning, dove l'ingegnerizzazione delle caratteristiche (*feature engineering*) è un passaggio cruciale e spesso

manuale, nel deep learning le rappresentazioni rilevanti vengono apprese automaticamente dal modello, direttamente dai dati grezzi. Questo rende il deep learning particolarmente potente in domini complessi come il riconoscimento di immagini, la traduzione automatica e l'elaborazione del linguaggio naturale.

Una delle architetture più efficaci nell'ambito del trattamento delle immagini è la **Convolutional Neural Network** (CNN). Queste reti sfruttano filtri convoluzionali che analizzano porzioni locali dell'immagine, apprendendo gerarchie di caratteristiche visive: dai bordi nei primi strati, fino a forme complesse e semantiche nei layer più profondi.

Il processo di addestramento di una rete neurale è un problema di ottimizzazione: l'obiettivo è trovare i pesi ottimali delle connessioni tra neuroni, tali da minimizzare una funzione obiettivo detta **funzione di perdita** (*loss function*). Questo avviene attraverso una tecnica chiamata *discesa del gradiente* (*Gradient Descent*), che aggiorna iterativamente i pesi nella direzione in cui riduce l'errore.

4.1.1 Backpropagation e Gradient Descent

Il meccanismo fondamentale che permette di calcolare l'errore in ciascun layer è noto come **Backpropagation**. Questo algoritmo calcola il gradiente della funzione di perdita rispetto ai pesi della rete, utilizzando la regola della catena (*chain rule*) del calcolo differenziale.

Il gradiente viene poi usato per aggiornare i pesi secondo la seguente formula:

$$w_{t+1} = w_t - \eta \cdot \nabla L(w_t)$$

dove w_t è il vettore dei pesi alla t -esima iterazione, η è il *learning rate*, e $\nabla L(w_t)$ è il gradiente della loss function rispetto ai pesi.

4.1.2 Attivazione Non Lineare

Ogni neurone in una rete applica una funzione di attivazione non lineare per aumentare la capacità espressiva del modello. Le funzioni più comuni includono:

- **ReLU** (Rectified Linear Unit): $f(x) = \max(0, x)$
- **Sigmoid**: $f(x) = \frac{1}{1+e^{-x}}$
- **Softmax**: $f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$

4.1.3 Overfitting e Generalizzazione

Un aspetto critico nel deep learning è il bilanciamento tra *apprendimento* e *generalizzazione*. Un modello può apprendere troppo bene i dati di training (fenomeno

noto come **overfitting**), perdendo la capacità di generalizzare su dati nuovi. Viceversa, un modello troppo semplice può non apprendere abbastanza (caso di **underfitting**).

Tecniche come la *regolarizzazione*, il *dropout*, la *data augmentation*, e una corretta suddivisione del dataset in training/validation/test sono essenziali per evitare questi problemi.

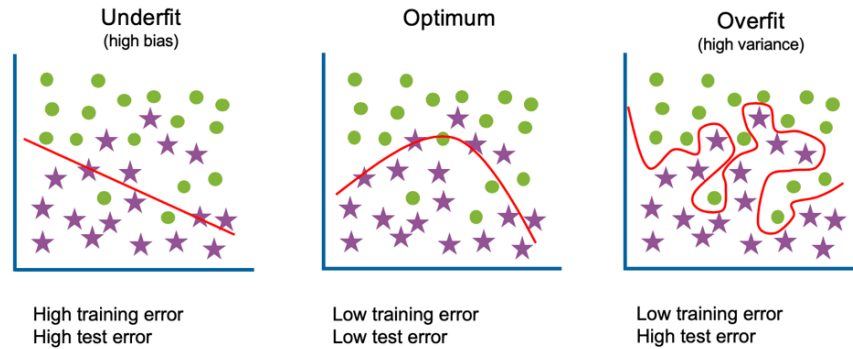


Figura 4.2: Esempio di overfitting, underfitting e buona generalizzazione [7]

4.1.4 Applicazioni del Deep Learning in Agricoltura

Negli ultimi anni, il settore agricolo ha beneficiato in modo significativo dell'applicazione delle tecnologie di deep learning, specialmente nell'ambito dell'agricoltura di precisione. Tra le sfide più rilevanti vi è la necessità di monitorare lo stato di salute delle coltivazioni in modo tempestivo e accurato. L'insorgenza di malattie nelle piante può compromettere interi raccolti e comportare gravi perdite economiche, specialmente se non identificata in tempo.

Il deep learning, grazie alla sua capacità di apprendere automaticamente rappresentazioni complesse a partire dai dati, si è rivelato estremamente efficace nel riconoscimento visivo di sintomi patologici sulle piante. Le reti neurali convoluzionali (CNN), in particolare, sono in grado di analizzare immagini di foglie, frutti o chiome degli alberi e identificare pattern che indicano la presenza di infezioni, stress o carenze nutritive. Tali pattern spesso sfuggono all'occhio umano o richiederebbero l'intervento di un esperto agronomo in campo, con costi e tempi molto elevati.

L'impiego di modelli di deep learning consente quindi un'analisi automatica, scalabile e non invasiva delle coltivazioni. Attraverso immagini acquisite da droni, satelliti o camere multispettrali montate su veicoli agricoli, è possibile ispezionare ampie superfici in tempi ridotti, identificando precocemente le aree colpite da malattie. Ciò permette agli agricoltori di intervenire in modo mirato, limitando

la diffusione dell'infezione e ottimizzando l'uso di pesticidi, fertilizzanti o risorse idriche.

Un ulteriore vantaggio dell'approccio automatico è la capacità di effettuare monitoraggi frequenti, garantendo una sorveglianza continua dello stato fitosanitario delle colture. Inoltre, grazie all'archiviazione storica dei dati e all'analisi predittiva, è possibile riconoscere schemi ricorrenti e prevenire future problematiche attraverso una gestione proattiva.

In sintesi, l'integrazione del deep learning nei processi agricoli rappresenta un'opportunità strategica per aumentare la produttività, ridurre gli sprechi e migliorare la sostenibilità dell'intero sistema agricolo. In questo lavoro, tale approccio viene applicato allo studio della salute degli alberi da frutto attraverso immagini multi-spettrali, con l'obiettivo di identificare automaticamente eventuali segni di malattia e supportare le decisioni agronomiche in modo efficiente.

4.2 Impostazione Iniziale e Riproduzione del Lavoro Precedente

Per impostare una base solida di confronto e verificare la validità del nuovo approccio, si è scelto inizialmente di replicare le configurazioni e i modelli utilizzati dal tesista precedente. Questa fase esplorativa ha come obiettivo principale quello di valutare l'impatto che la riduzione del dataset ha sulle prestazioni dei modelli. In particolare, si intende osservare se l'eliminazione di una porzione del dataset comporti un degrado significativo in termini di accuratezza e capacità predittiva.

4.2.1 Tecniche Utilizzate: Transfer Learning e Data Augmentation

Le reti neurali convoluzionali (CNN) richiedono generalmente una grande quantità di dati per poter apprendere rappresentazioni efficaci. Per mitigare il problema della scarsità di dati — acuito dalla riduzione del dataset rispetto al lavoro precedente — sono state adottate due tecniche fondamentali:

Transfer Learning. Sono stati utilizzati modelli pre-addestrati su grandi dataset (in questo caso ImageNet), in modo da sfruttare le conoscenze apprese su compiti generici per il nostro specifico problema. I primi strati della rete vengono inizialmente congelati e solo gli strati finali sono addestrati sui dati agricoli, riducendo il rischio di overfitting e velocizzando l'apprendimento.

Data Augmentation. Per aumentare artificialmente la dimensione del dataset e migliorarne la diversità, sono state applicate trasformazioni geometriche alle

immagini originali, tra cui rotazioni, riflessioni (flip), zoom e traslazioni. Questo processo aiuta a rendere il modello più robusto rispetto alle variazioni naturali che possono verificarsi nelle immagini acquisite sul campo.

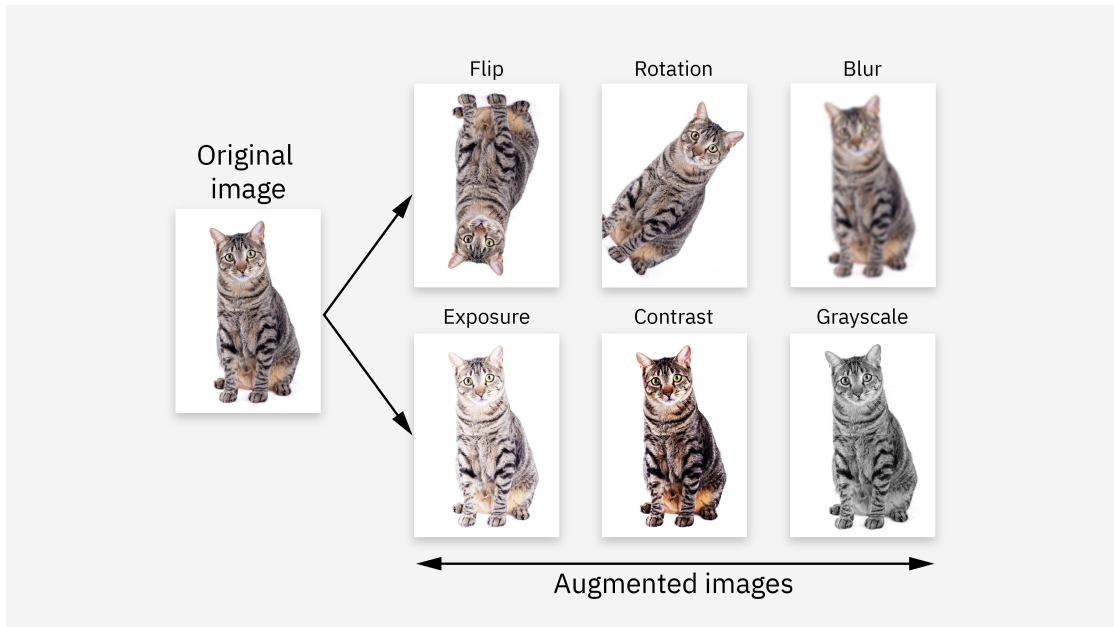


Figura 4.3: Esempio di *data augmentation*: immagine originale e sue trasformazioni [8].

4.2.2 Architetture Scelte

Le architetture selezionate per questa prima fase sono **ResNet50** e **DenseNet121**, disponibili nella libreria **Keras Applications**. Entrambe rappresentano soluzioni consolidate per compiti di classificazione di immagini e sono state preferite per l'equilibrio tra efficienza computazionale e accuratezza.

La scelta degli iperparametri — come *batch size*, *learning rate* e *numero di epoche* — è stata effettuata manualmente, tenendo conto delle limitazioni hardware e del tempo disponibile. Non è stato pertanto possibile impiegare tecniche più avanzate di ottimizzazione automatica come *grid search* o *random search*.

Layer (type)	Output Shape	Param #
input_12 (InputLayer)	[(None, 224, 224, 3)]	0
resnet50 (Functional)	(None, 7, 7, 2048)	23587712
global_average_pooling2d_3 (GlobalAveragePooling2D)	(None, 2048)	0
dense_18 (Dense)	(None, 1000)	2049000
dropout (Dropout)	(None, 1000)	0
dense_19 (Dense)	(None, 500)	500500
dropout_1 (Dropout)	(None, 500)	0
dense_20 (Dense)	(None, 500)	250500
dropout_2 (Dropout)	(None, 500)	0
dense_21 (Dense)	(None, 100)	50100
dense_22 (Dense)	(None, 16)	1616
dense_23 (Dense)	(None, 1)	17
=====		
Total params: 26,439,445		
Trainable params: 3,906,453		
Non-trainable params: 22,532,992		

Figura 4.4: Struttura dell'architettura ResNet50.

Layer (type)	Output Shape	Param #
input_9 (InputLayer)	[(None, 224, 224, 3)]	0
densenet121 (Functional)	(None, 7, 7, 1024)	7037504
global_average_pooling2d_2 (GlobalAveragePooling2D)	(None, 1024)	0
dense_12 (Dense)	(None, 1000)	1025000
dropout_8 (Dropout)	(None, 1000)	0
dense_13 (Dense)	(None, 500)	500500
dropout_9 (Dropout)	(None, 500)	0
dense_14 (Dense)	(None, 500)	250500
dropout_10 (Dropout)	(None, 500)	0
dense_15 (Dense)	(None, 100)	50100
dropout_11 (Dropout)	(None, 100)	0
dense_16 (Dense)	(None, 16)	1616
dense_17 (Dense)	(None, 1)	17
=====		
Total params: 8,865,237		
Trainable params: 1,827,733		
Non-trainable params: 7,037,504		

Figura 4.5: Struttura dell'architettura DenseNet121.

4.2.3 Strategia di Addestramento

L'addestramento di una rete neurale profonda è un processo iterativo di ottimizzazione che mira a migliorare progressivamente le prestazioni del modello attraverso l'aggiornamento dei pesi sinaptici. All'inizio dell'addestramento, i pesi sono inizializzati casualmente. Successivamente, vengono ottimizzati minimizzando una funzione di perdita (loss function) che quantifica la distanza tra le predizioni del modello e i valori reali.

Il dataset è stato suddiviso in tre sottoinsiemi secondo le stesse proporzioni utilizzate nel lavoro precedente:

- **Training set:** 70%
- **Validation set:** 15%
- **Test set:** 15%

Per l'ottimizzazione è stato adottato l'algoritmo *Stochastic Gradient Descent* (SGD), che aggiorna i pesi dopo ogni mini-batch di dati. Nei primi esperimenti, è stata utilizzata la funzione di perdita *Mean Squared Error* (MSE), in quanto il problema iniziale è stato formulato come una regressione tra i valori di probabilità associati alle classi (albero sano o malato). La MSE misura la media dei quadrati delle differenze tra le predizioni del modello e i valori attesi:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (4.1)$$

dove:

- n è il numero totale di campioni,
- y_i è il valore reale del campione i -esimo,
- $\hat{y}_i$ è il valore predetto dal modello per il campione i -esimo.

La MSE è particolarmente utile quando si desidera penalizzare fortemente errori di classificazione con valori di output molto distanti dalla verità.

La scelta della MSE in questa fase iniziale è stata guidata da una necessità di mantenere coerenza con gli esperimenti condotti dal tesista precedente, facilitando un confronto diretto tra i risultati. In fasi successive, sarà possibile valutare l'impatto dell'utilizzo di funzioni di perdita diverse, più specifiche per la classificazione binaria, come la *binary cross-entropy*.

Il processo di addestramento è stato strutturato in due fasi consecutive, per un totale di 100 epoche, adottando una strategia di *fine-tuning* che ha permesso di combinare la generalizzazione di un modello pre-addestrato con l'adattamento ai dati agricoli specifici di questo studio.

- **Fase 1 – Addestramento dei layer finali (epoche 1–50):** nella prima metà del training, i pesi della *backbone* pre-addestrata (ResNet50 o DenseNet121) sono stati mantenuti congelati. In questa fase, soltanto i layer fully connected aggiunti in coda alla rete sono stati aggiornati. Questo consente al modello di imparare a interpretare le caratteristiche già estratte dai layer convoluzionali senza alterarne la struttura.

- **Fase 2 – Fine-tuning mirato (epoche 51–100):** nella seconda metà del processo di training, oltre ai layer fully connected, sono stati sbloccati anche alcuni degli ultimi strati convoluzionali del modello pre-addestrato. Questo approccio di fine-tuning mirato ha permesso di aggiornare solo le parti finali della rete, responsabili delle rappresentazioni di alto livello, migliorando l'adattamento al dominio agricolo delle immagini multispettrali senza compromettere la stabilità dell'apprendimento.

4.2.4 Risultati Preliminari

I risultati ottenuti in questa prima fase, nonostante la riduzione del dataset rispetto a quello utilizzato nel lavoro precedente, mostrano una coerenza sorprendente con le prestazioni precedentemente documentate. In particolare, la rete **ResNet50** ha evidenziato una superiorità in termini di accuratezza complessiva rispetto alla **DenseNet121**, confermandosi come l'architettura più promettente tra quelle sperimentate.

Il modello ha raggiunto un'accuratezza intorno al **70%**, un risultato particolarmente significativo se confrontato con il valore di **71.8%** ottenuto dal testista precedente. Questo margine minimo di differenza suggerisce che, pur operando in condizioni meno favorevoli per quanto riguarda la disponibilità di dati, sia possibile mantenere un buon livello di generalizzazione.

Tali performance confermano non solo la solidità intrinseca delle architetture adottate, ma anche l'efficacia delle tecniche di *transfer learning* e *data augmentation* nell'adattare modelli pre-addestrati a contesti specifici come quello agricolo.

Questi risultati preliminari costituiscono dunque una base solida su cui costruire ulteriori esperimenti. Nelle sezioni successive verranno esplorate nuove architetture e strategie di training, con l'obiettivo di migliorare ulteriormente le prestazioni del sistema di classificazione automatica delle chiome.

4.3 Modifiche e Prove di nuove architetture

Nel corso delle sperimentazioni sono state introdotte alcune modifiche significative nel setup del training, con l'obiettivo di migliorare la capacità di generalizzazione dei modelli. Una delle prime variazioni ha riguardato l'estensione delle tecniche di *data augmentation*, che risultano fondamentali in scenari con dataset limitati e in presenza di variabilità nella fase di acquisizione.

In particolare, rispetto agli esperimenti precedenti, sono state aggiunte:

- rotazioni casuali con un angolo massimo di 30° ,
- variazioni di luminosità (*brightness*) con un intervallo da 0.8 a 1.2,

- traslazioni orizzontali e verticali (*width* e *height shift*).

L'introduzione di queste trasformazioni ha lo scopo di aumentare l'invarianza del modello rispetto a rotazioni, traslazioni e variazioni di scala, migliorandone la robustezza in fase di test. Questo approccio è particolarmente utile in contesti in cui le immagini sono acquisite da UAV (Unmanned Aerial Vehicles), e dunque soggette a variazioni di prospettiva, orientamento e illuminazione.

4.3.1 Modifiche architetturali e strategie di training

Per tutte le architetture sperimentate, sono stati caricati i pesi pre-addestrati su ImageNet, sfruttando quindi il *transfer learning* per migliorare la convergenza e le prestazioni del modello anche su un dataset di dimensioni limitate. La parte finale della rete, ovvero i *fully connected layers*, è stata modificata rimuovendo i livelli originari e inserendo una nuova sequenza di layer densi intervallati da funzioni di attivazione ReLU, fino a ottenere un output di due unità. Su quest'ultimo viene applicata la funzione di attivazione SoftMax, in modo da restituire, per ciascuna immagine in input, due valori corrispondenti alla probabilità che la porzione di nocciolo nell'immagine appartenga alla classe "sano" o "malato".

I modelli sono stati compilati utilizzando l'ottimizzatore SGD (Stochastic Gradient Descent) con un *learning rate* iniziale di 0.01 e un *momentum* pari a 0.9.

Il termine *momentum* rappresenta un'accelerazione direzionale che aiuta a mantenere il verso della discesa nei minimi locali o in regioni piatte della funzione di perdita, contribuendo a una convergenza più stabile ed efficiente. Formalmente, l'aggiornamento dei pesi con momentum si esprime come:

$$v_t = \gamma v_{t-1} + \eta \nabla J(\theta_{t-1}) \quad (4.2)$$

$$\theta_t = \theta_{t-1} - v_t \quad (4.3)$$

dove:

- θ_t rappresenta i pesi al tempo t ,
- η è il *learning rate*,
- γ è il *momentum* (in questo caso 0.9),
- $\nabla J(\theta_{t-1})$ è il gradiente della funzione di perdita rispetto ai pesi,
- v_t è la "velocità" dell'aggiornamento.

La loss function adottata è la **categorical crossentropy**, appropriata per problemi di classificazione multiclasse (in questo caso due classi: albero sano vs. albero malato). Essa misura la distanza tra la distribuzione di probabilità predetta dal modello e la distribuzione reale (target) ed è definita come:

$$\mathcal{L} = - \sum_{i=1}^C y_i \log(\hat{y}_i) \quad (4.4)$$

dove:

- C è il numero di classi (in questo caso 2),
- y_i è la probabilità vera della classe i (solitamente 0 o 1),
- $\hat{y}_i$ è la probabilità predetta dal modello per la classe i .

Durante l'addestramento viene monitorata la *validation loss*, e nel caso in cui essa non mostri miglioramenti per 5 epoche consecutive, il *learning rate* viene ridotto del 50% (fino a un minimo di $1 \cdot 10^{-6}$). Questa strategia permette al modello di effettuare aggiornamenti più piccoli nelle fasi finali dell'addestramento, favorendo una migliore esplorazione del minimo locale della funzione di perdita.

Il processo di training si articola in due fasi, come già effettuato negli esperimenti precedenti:

1. una prima fase di 50 epoche in cui viene congelata tutta la parte convoluzionale della rete, allenando esclusivamente i nuovi layer fully connected;
2. una seconda fase di ulteriori 50 epoche in cui viene "scongelata" la porzione finale della parte convoluzionale (circa il 25-30% dei layer), permettendo anche a questi di aggiornare i propri pesi durante l'ottimizzazione.

Questo approccio graduale consente di preservare le rappresentazioni di basso livello apprese su ImageNet, adattando progressivamente i livelli più astratti al nuovo task specifico di classificazione delle immagini dei noccioli.

Per gli strati *fully connected* (FC) è stata utilizzata la strategia di inizializzazione dei pesi nota come **Xavier Uniform Initialization**. Questo metodo mira a mantenere costante la *varianza dei segnali* che si propagano attraverso i vari strati della rete neurale, riducendo il rischio di *scomparsa* (*vanishing*) o *esplosione* (*exploding*) del gradiente durante il processo di *backpropagation*.

L'idea di base è che i pesi vengano inizializzati in modo tale che la varianza dell'attivazione sia la stessa in ingresso e in uscita da ciascun layer. In particolare, per un layer con n_{in} neuroni in ingresso e n_{out} in uscita, i pesi W vengono estratti da una distribuzione uniforme definita come:

$$W \sim \mathcal{U}\left(-\sqrt{\frac{6}{n_{\text{in}} + n_{\text{out}}}}, \sqrt{\frac{6}{n_{\text{in}} + n_{\text{out}}}}\right)$$

Anche se la Xavier Initialization è stata originariamente proposta per funzioni di attivazione simmetriche (come tanh o sigmoid), è stata comunque utilizzata in questo lavoro anche in presenza di attivazioni di tipo ReLU, dato il numero limitato di questi layer (2–3); inoltre, ha fornito buoni risultati empirici e ha garantito stabilità durante l'addestramento.

4.3.2 Architetture testate

Per valutare l'efficacia di diverse reti neural network pre-addestrate, sono stati esplorati i seguenti modelli: **VGG16**, **VGG19**, **ResNet50**, **InceptionV3**, **EfficientNetB3** e **EfficientNetB5**. Tutte le reti sono state adattate al compito di classificazione binaria tramite la sostituzione dei livelli MLP finali.

VGG16 e VGG19

Le reti VGG16 e VGG19 sono due varianti della famiglia VGG (Visual Geometry Group), note per la loro struttura semplice. Entrambe le architetture si basano su una sequenza di blocchi convoluzionali con kernel di dimensione 3×3 e operazioni di *max pooling*. La principale differenza tra le due risiede nella profondità: VGG16 ha 13 strati convoluzionali, mentre VGG19 ne ha 16, grazie all'aggiunta di tre convoluzioni nei blocchi intermedi.

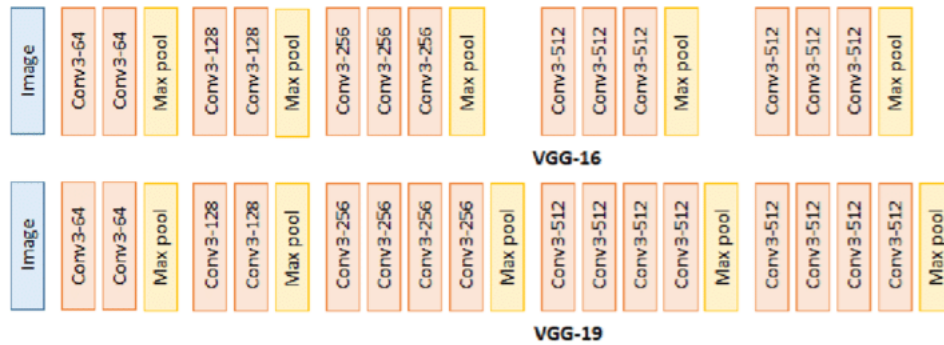


Figura 4.6: Architetture di VGG16 e VGG19 [9].

Model: "VGG16 Model"

Layer (type)	Output Shape	Param #
Input Image	(None, 224, 224, 3)	0
vgg16 (Functional)	(None, 7, 7, 512)	14,714,688
flatten (Flatten)	(None, 25088)	0
FC1_ReLU (Dense)	(None, 4192)	105,173,088
FC2_ReLU (Dense)	(None, 512)	2,146,816
FC3_Softmax (Dense)	(None, 2)	1,026

Total params: 122,035,618 (465.53 MB)
 Trainable params: 107,320,930 (409.40 MB)
 Non-trainable params: 14,714,688 (56.13 MB)

Figura 4.7: Modello VGG16 trainato

Model: "VGG19 Model"

Layer (type)	Output Shape	Param #
Input Image	(None, 224, 224, 3)	0
vgg19 (Functional)	(None, 7, 7, 512)	20,024,384
flatten (Flatten)	(None, 25088)	0
FC1_ReLU (Dense)	(None, 4192)	105,173,088
FC2_ReLU (Dense)	(None, 512)	2,146,816
FC3_Softmax (Dense)	(None, 2)	1,026

Total params: 127,345,314 (485.78 MB)
 Trainable params: 107,320,930 (409.40 MB)
 Non-trainable params: 20,024,384 (76.39 MB)

Figura 4.8: Modello VGG19 trainato

Inception e ResNet

Oltre alle reti VGG, si è scelto di testare architetture più profonde e articolate; per questo, sono state scelte le reti Inception e ResNet, con l'obiettivo di valutare se una maggiore complessità strutturale potesse migliorare le prestazioni nella classificazione degli alberi di nocciolo.

La rete **Inception** si distingue per l'utilizzo di moduli "inception", che combinano convoluzioni con diversi kernel (ad esempio 1×1 , 3×3 , 5×5) all'interno dello stesso blocco. Questo approccio permette di catturare informazioni a diverse scale e ridurre il numero complessivo di parametri, migliorando l'efficienza della rete.

La rete **ResNet** (Residual Network), invece, è caratterizzata dalla presenza di *connessioni residue*, ovvero collegamenti diretti tra blocchi di convoluzione non consecutivi. Questi shortcut consentono di superare il problema del degrado delle prestazioni in reti molto profonde, facilitando la propagazione del gradiente durante l'addestramento.

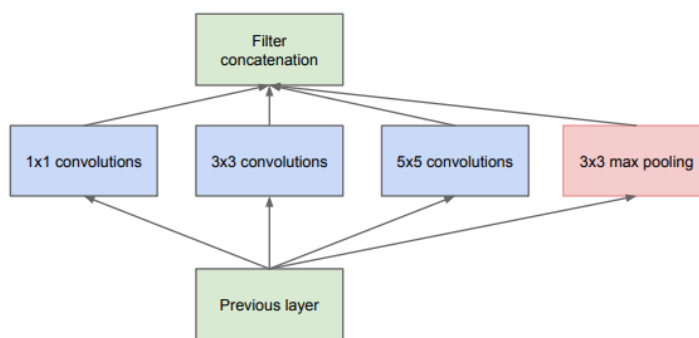


Figura 4.9: Schema di un Inception block [10].

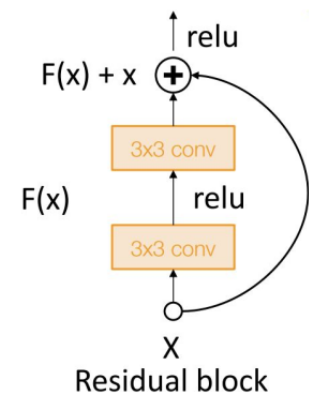


Figura 4.10: Esempio di un residual block.

Model: "InceptionV3 Model"

Layer (type)	Output Shape	Param #
Input Image	(None, 224, 224, 3)	0
InceptionV3 (Functional)	(None, 5, 5, 2048)	21,802,784
Global Average Pooling (GlobalAveragePooling2D)	(None, 2048)	0
FC1_ReLU (Dense)	(None, 2048)	4,196,352
Dropout1 (Dropout)	(None, 2048)	0
FC2_ReLU (Dense)	(None, 512)	1,049,088
Dropout2 (Dropout)	(None, 512)	0
FC3_Softmax (Dense)	(None, 2)	1,026

Total params: 27,049,250 (103.18 MB)
 Trainable params: 5,246,466 (20.01 MB)
 Non-trainable params: 21,802,784 (83.17 MB)

Figura 4.11: Modello Inception trainato.

Model: "ResNet50 Model"

Layer (type)	Output Shape	Param #
Input Image	(None, 224, 224, 3)	0
resnet50 (Functional)	(None, 7, 7, 2048)	23,587,712
Global Average Pooling (GlobalAveragePooling2D)	(None, 2048)	0
FC1_ReLU (Dense)	(None, 1024)	2,098,176
FC2_ReLU (Dense)	(None, 256)	262,400
FC3_Softmax (Dense)	(None, 2)	514

Total params: 25,948,802 (98.99 MB)
 Trainable params: 2,361,090 (9.01 MB)
 Non-trainable params: 23,587,712 (89.98 MB)

Figura 4.12: Modello ResNet trainato.

EfficientNet

Oltre alle architetture già analizzate, è stato sperimentato anche il modello EfficientNet, noto per il bilanciamento efficace tra profondità, larghezza e risoluzione, e adottato anche nell'articolo *Plant Leaf Disease Classification Using EfficientNet Deep Learning Model* [11], dove ha mostrato buone prestazioni nella classificazione automatica delle malattie fogliari. EfficientNet si basa su un approccio innovativo di *scaling* della rete, denominato *compound scaling*, che consente di bilanciare congiuntamente profondità, larghezza e risoluzione dell'immagine attraverso un unico parametro di scala [12].

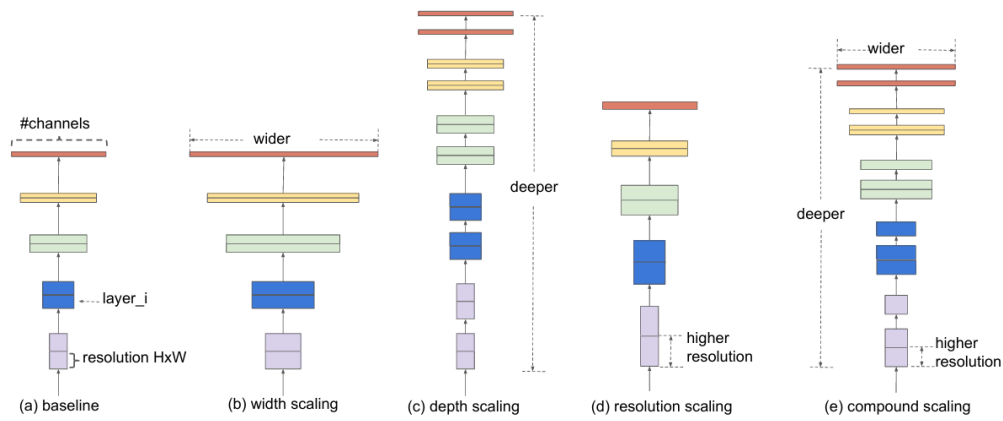


Figura 4.13: (a) rappresenta una rete di base; (b), (c) e (d) illustrano l'incremento isolato rispettivamente di larghezza, profondità e risoluzione. (e) mostra il *compound scaling*, che scala simultaneamente tutte e tre le dimensioni seguendo un rapporto fisso [12].

Sono state testate due versioni della rete:

- **EfficientNetB3**, con input dimensione 300×300 ;
- **EfficientNetB5**, con input dimensione 456×456 .

L'obiettivo era valutare se l'aumento della dimensione dell'immagine in input potesse migliorare le prestazioni nella classificazione delle chiome di nocciolo. EfficientNet è infatti progettata per scalare bene con immagini ad alta risoluzione, e rappresenta quindi una scelta interessante per il contesto dell'agricoltura di precisione, dove i dettagli morfologici sono spesso sottili e rilevabili solo a risoluzioni elevate.

Model: "EfficientNetB3 Model"

Layer (type)	Output Shape	Param #
Input Image	(None, 300, 300, 3)	0
EfficientNetB3 (Functional)	(None, 10, 10, 1536)	10,783,535
Global Average Pooling (GlobalAveragePooling2D)	(None, 1536)	0
FC1_ReLU (Dense)	(None, 1024)	1,573,888
FC2_ReLU (Dense)	(None, 256)	262,400
FC3_ReLU (Dense)	(None, 64)	16,448
FC3_Softmax (Dense)	(None, 2)	130

Total params: 12,636,401 (48.20 MB)
 Trainable params: 1,852,866 (7.07 MB)
 Non-trainable params: 10,783,535 (41.14 MB)

Figura 4.14: Modello EfficientNetB3 trainato.

Model: "EfficientNetB5 Model"

Layer (type)	Output Shape	Param #
Input Image	(None, 456, 456, 3)	0
EfficientNetB5 (Functional)	(None, 15, 15, 2048)	28,513,527
Global Average Pooling (GlobalAveragePooling2D)	(None, 2048)	0
FC1_ReLU (Dense)	(None, 1024)	2,098,176
FC2_ReLU (Dense)	(None, 256)	262,400
FC3_ReLU (Dense)	(None, 64)	16,448
FC3_Softmax (Dense)	(None, 2)	130

Total params: 30,890,681 (117.84 MB)
 Trainable params: 2,377,154 (9.07 MB)
 Non-trainable params: 28,513,527 (108.77 MB)

Figura 4.15: Modello EfficientNetB5 trainato.

4.3.3 Risultati

Dopo l'addestramento dei modelli descritti in precedenza, i risultati sono stati valutati sulla base delle **matrici di confusione** e delle metriche derivate: **accuratezza**, **precisione** e **recall**, con particolare attenzione alla **classe 0** (piante malate), che rappresenta il principale interesse applicativo in ambito agricolo.

Nota sul training prolungato: per tutti i modelli è stato effettuato un ulteriore ciclo di addestramento di 50 epoche, al fine di verificare se un training più esteso potesse migliorare le prestazioni. In generale, l'aumento del numero di epoche non ha portato a benefici significativi, con l'eccezione del modello **ResNet50**. In questo caso, estendendo l'addestramento, si è ottenuti dei miglioramenti nella predizione degli alberi malati a discapito di quelli sani.

Matrici di confusione

Di seguito sono riportate le confusion matrix per ciascun modello. Per ResNet50 sono mostrati sia i risultati ottenuti con il training standard, sia quelli dopo un addestramento prolungato.

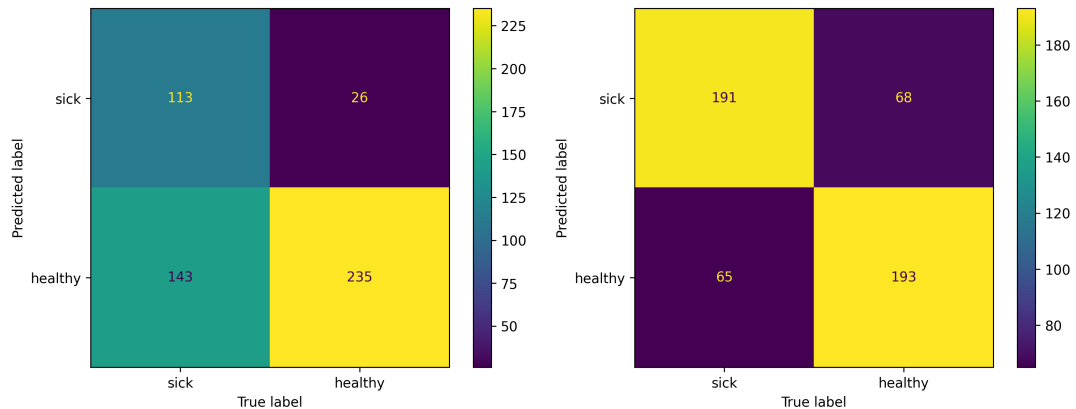


Figura 4.16: Matrici di confusione per i modelli VGG16 (sinistra) e VGG19 (destra).

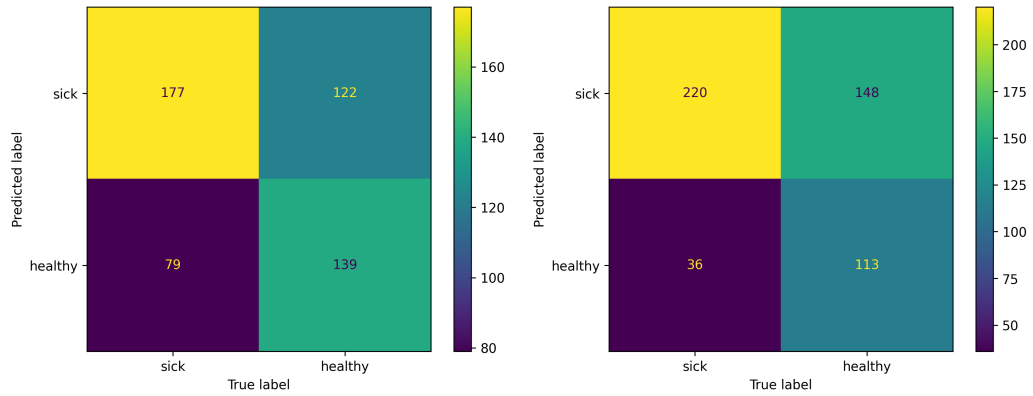


Figura 4.17: Matrici di confusione per il modello ResNet50 con training standard (sinistra) e con addestramento prolungato (destra).

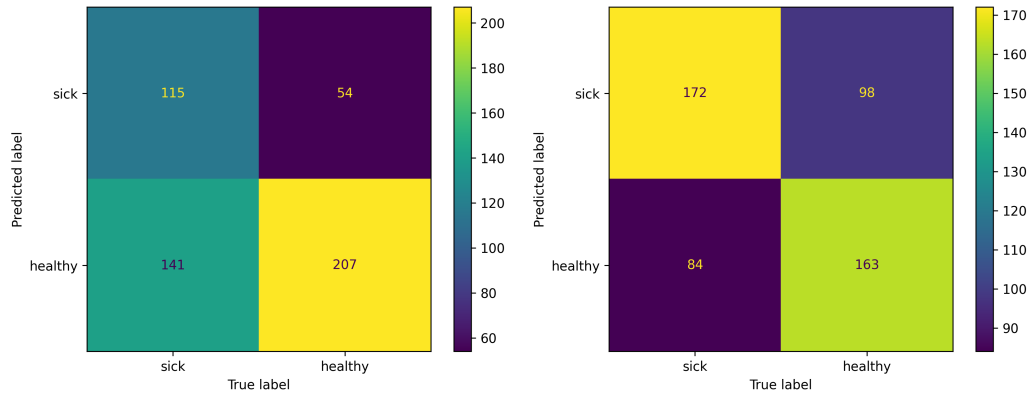


Figura 4.18: Matrici di confusione per i modelli EfficientNetB3 (sinistra) e EfficientNetB5 (destra).

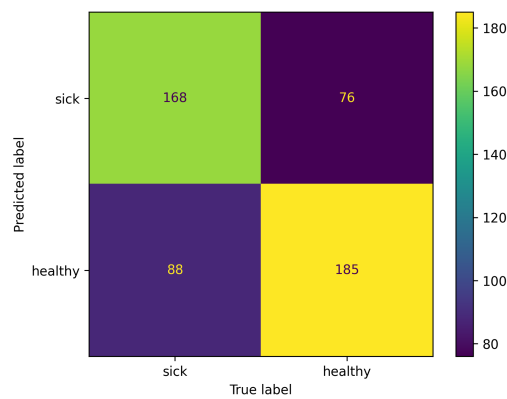


Figura 4.19: Matrici di confusione per il modello InceptionV3.

Dall'analisi delle matrici di confusione emergono osservazioni significative riguardo alla capacità dei modelli di riconoscere correttamente le piante malate. In questo contesto, i True Positive (TP) rappresentano le piante effettivamente malate che sono state correttamente classificate come tali, mentre i False Positive (FP) indicano le piante sane che sono state erroneamente classificate come malate. Il modello VGG19 si distingue per il numero più alto di True Positive (191) e un numero relativamente contenuto di False Negative (65), dimostrando una buona capacità nel riconoscere correttamente le piante malate. Anche EfficientNetB5 presenta performance simili, con 172 TP e 84 FN, confermando un buon equilibrio tra sensibilità e accuratezza.

Al contrario, VGG16 ed EfficientNetB3 ottengono un numero molto basso di True Positive (rispettivamente 113 e 115), con un numero elevato di False Negative (143 e 141), indicando difficoltà nel riconoscere tutti i casi malati. InceptionV3 si colloca in una posizione intermedia, con 168 TP e 88 FN, offrendo un compromesso accettabile.

Per quanto riguarda ResNet50, si osserva un comportamento interessante tra l'addestramento standard e quello con più epoche. In particolare, l'aumento delle epoche comporta un miglioramento dei True Positive, che passano da 177 a 220, e una riduzione dei False Negative da 79 a 36. Tuttavia, questo avviene a scapito dei True Negative, che scendono da 139 a 113, e con un aumento dei False Positive. Dal nostro punto di vista, l'incremento nel riconoscimento delle piante malate (TP) è prioritario rispetto al contenimento dei falsi allarmi, rendendo l'addestramento prolungato una scelta preferibile in questo contesto applicativo.

Metriche di valutazione

Per ciascun modello sono state calcolate le seguenti metriche:

- **Accuracy:** proporzione di predizioni corrette rispetto al totale delle osservazioni.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision:** percentuale di istanze classificate come malate che sono effettivamente malate. Misura quanto siano affidabili le predizioni positive del modello.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall:** percentuale di piante malate correttamente individuate. Su tutte le piante malate nel dataset, quante sono state effettivamente trovate dal modello?

$$\text{Recall} = \frac{TP}{TP + FN}$$

Queste metriche, calcolate sulla classe 1 (piante malate), permettono di comprendere il comportamento dei modelli in un contesto in cui il rilevamento dei casi positivi è cruciale.

Analisi dei risultati

Tabella 4.1: Metriche di performance per ciascun modello

Modello	Accuracy (%)	Precision (%)	Recall (%)
VGG16	67.3	81.3	44.1
VGG19	74.3	73.7	74.6
ResNet50	64.4	59.8	85.9
InceptionV3	68.3	68.9	65.6
EfficientNetB3	62.3	68.0	44.9
EfficientNetB5	64.8	63.7	67.2

Dalla tabella emerge un quadro differenziato delle prestazioni dei modelli.

Il modello VGG19 risulta il più bilanciato e per quanto riguarda l'accuratezza il migliore, con una accuracy complessiva del 74.3%, una precision pari a 73,7% e una recall di 74.6%. Ciò indica che è in grado sia di individuare correttamente un buon numero di piante malate, sia di non eccedere nei falsi positivi: una prestazione equilibrata e robusta.

ResNet50 si distingue per la recall più alta (85.9%), ovvero per la capacità di rilevare la maggior parte delle piante malate, anche se a scapito della precisione (59.8%). Questo significa che il modello tende a segnalare molte piante come malate, anche quando non lo sono, ma riesce comunque a coprire la maggior parte dei casi reali di malattia. È quindi adatto a contesti dove la priorità è non perdere casi malati, anche accettando qualche falso allarme.

VGG16, pur avendo una buona precision (81.3%), mostra una recall molto bassa (44.1%). Questo indica un comportamento conservativo: segnala una pianta come malata solo quando è molto sicuro, ma si lascia sfuggire molti casi reali di malattia. In termini applicativi, potrebbe non essere adatto se l'obiettivo principale è l'individuazione precoce delle piante infette.

InceptionV3 e EfficientNetB5 si collocano in una zona intermedia, con valori di precision e recall piuttosto bilanciati ma non particolarmente elevati.

Infine, EfficientNetB3 presenta sia precision (68.0%) che recall (44.9%) relativamente basse, e anche l'accuracy è la più bassa tra i modelli (62.3%), suggerendo una minore efficacia generale in questo contesto.

Conclusioni

Nel complesso, la rete **VGG19** si è dimostrata l'architettura più efficace, con il miglior compromesso tra accuratezza, precisione e recall. Nonostante la crescente complessità e modernità di modelli come ResNet e EfficientNet, nel contesto specifico della classificazione delle piante di nocciolo malate, un'architettura più semplice ma consolidata come VGG19 si è dimostrata più adatta.

4.4 Estensione all'analisi multispettrale

Dopo la fase di sperimentazione condotta su immagini RGB, si è scelto di esplorare l'impiego di dati multispettrali, con l'obiettivo di valutare se bande non visibili come il *red-edge* o il *near-infrared* potessero offrire un contributo significativo nel riconoscimento delle piante malate.

Questa scelta è stata motivata sia da considerazioni teoriche – in particolare, il fatto che i sintomi di stress o malattia vegetale possono manifestarsi inizialmente a livello fisiologico, prima ancora che siano visibili nel dominio RGB – sia da precedenti studi in letteratura. In particolare, il lavoro *Detection of Bark Beetle Disturbance at Tree Level Using UAS Multispectral Imagery and Deep Learning* ha evidenziato l'efficacia delle immagini multispettrali e delle reti neurali profonde nel rilevare anomalie a livello arboreo.

Le informazioni contenute nelle bande spettrali aggiuntive risultano infatti particolarmente sensibili alla variazione della struttura fogliare, del contenuto d'acqua e della clorofilla, rendendole potenzialmente più efficaci per la rilevazione precoce di anomalie.

L'analisi condotta con le immagini multispettrali mira quindi a verificare se tale ricchezza informativa possa tradursi in un miglioramento delle capacità predittive dei modelli, soprattutto in termini di sensibilità nei confronti delle piante affette da malattia.

Il setup sperimentale per l'elaborazione delle immagini multispettrali è stato impostato in modo analogo a quello utilizzato per le immagini RGB. Poiché i file **TIF** a disposizione contenevano cinque bande spettrali distinte, mentre le reti neurali impiegate accettano in ingresso immagini a tre canali, è stato necessario sperimentare diverse combinazioni di bande. A tal fine, si è fatto riferimento al modello che aveva ottenuto le migliori prestazioni nella fase RGB, ovvero la rete **VGG19**, limitando l'analisi a tale architettura per motivi di tempo e per evitare un'estensione eccessiva di questa fase, dato che l'addestramento di ciascun modello richiede diverse ore.

Sono quindi state testate varie combinazioni di bande spettrali, e la configurazione che ha restituito prestazioni migliori è risultata essere quella composta dalle bande *red*, *green* e *blue*.

4.4.1 Adattamenti al setup di training per immagini multispettrali

Durante la fase iniziale di addestramento con le immagini multispettrali, sono emerse due problematiche principali: da un lato, un evidente **overfitting**, e dall'altro una **scarsa capacità di apprendimento da parte del modello**, che spesso non superava un'accuratezza del 50–55%.

L'overfitting si manifestava con una rapida riduzione della *training loss*, mentre la *validation loss* rimaneva pressoché invariata. In altri casi, invece, il modello mostrava difficoltà a imparare qualunque pattern significativo, convergendo velocemente a una previsione quasi casuale.

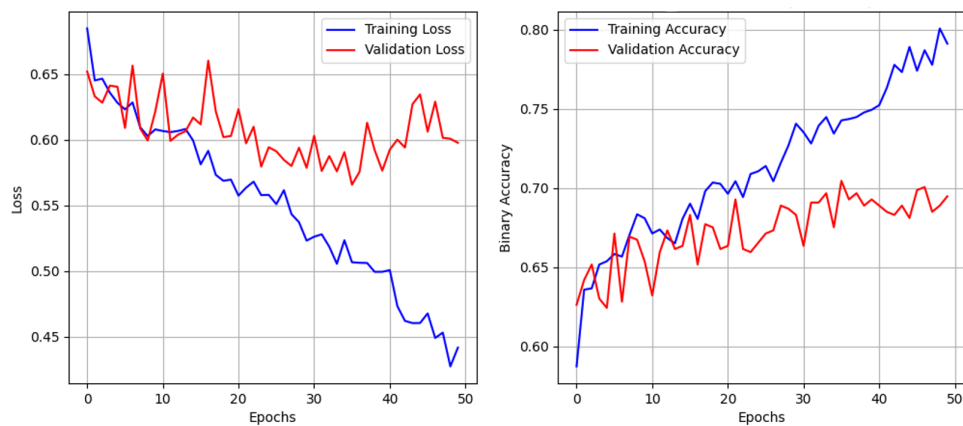


Figura 4.20: Andamento della *loss* (sinistra) e dell'*accuracy* (destra) durante la seconda fase di training del modello **VGG19**, in cui viene scongelata una parte della rete convoluzionale. È evidente la presenza di overfitting, con la *training accuracy* che continua a migliorare, mentre la *validation accuracy* si stabilizza attorno al 70%.

Per far fronte a questi problemi, è stato necessario modificare il setup originariamente utilizzato per le immagini RGB:

- **Momentum:** il valore del parametro è stato ridotto da 0,9 a 0,5. Tale modifica si è rivelata cruciale per migliorare la convergenza, poiché valori troppo elevati impedivano al modello di aggiornare efficacemente i pesi, portando a una stagnazione dell'apprendimento.
- **Data Augmentation:** è stata rimossa la trasformazione di *brightness modification*. Mentre questa tecnica si è dimostrata utile con immagini RGB, nel contesto multispettrale ha introdotto variazioni poco realistiche tra le bande,

ostacolando la fase di apprendimento. La sua eliminazione ha contribuito a un netto miglioramento dell'accuratezza.

- **Regolarizzazione con Dropout:** per contenere il fenomeno di overfitting, è stata testata l'introduzione di più layer di *dropout* nelle fasi finali (*fully connected*) della rete. L'effetto è stato coerente: la presenza del dropout ha effettivamente ridotto la discrepanza tra training e validation accuracy, tuttavia spesso a scapito della performance complessiva, con una leggera riduzione dell'accuratezza.

Per questo motivo, si è deciso di allenare sia versioni con dropout che versioni senza, confrontando poi direttamente i risultati per valutare il compromesso ottimale tra generalizzazione e accuratezza.

Questi adattamenti dimostrano come l'elaborazione di immagini multispettrali richieda una personalizzazione attenta della pipeline di training, soprattutto per quanto riguarda le strategie di ottimizzazione e regolarizzazione. Le osservazioni condotte suggeriscono che, pur partendo da architetture standard pre-addestrate su RGB, un tuning mirato degli iperparametri sia essenziale per sfruttare appieno le potenzialità dei dati multispettrali.

Matrici di confusione

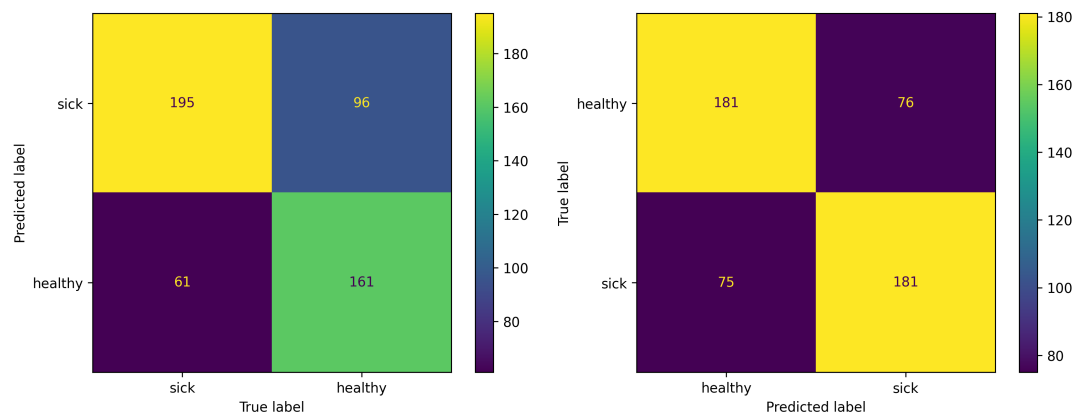


Figura 4.21: Matrici di confusione per i modelli VGG16 (sinistra) e VGG19 (destra).

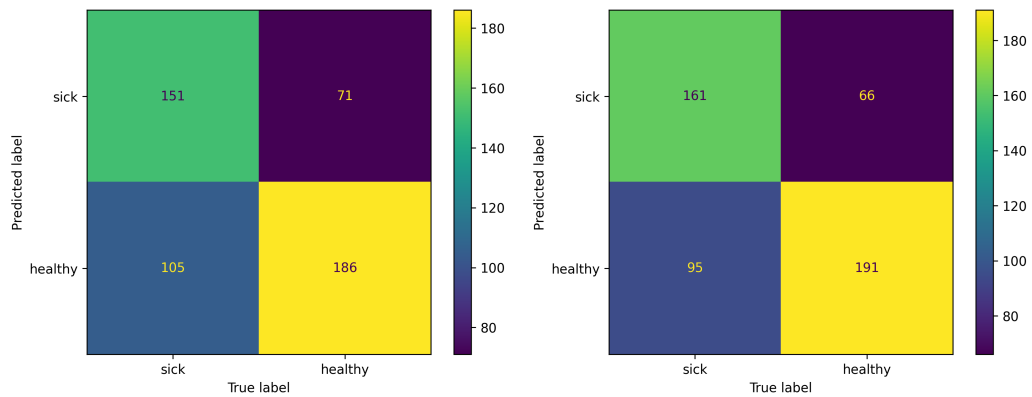


Figura 4.22: Matrice di confusione per i modelli ResNet50 (sinistra) e InceptionV3 (destra).

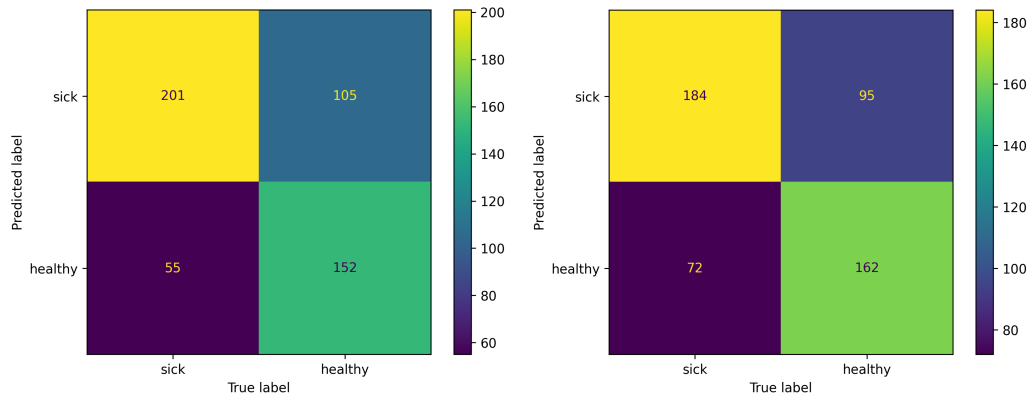


Figura 4.23: Matrici di confusione per i modelli EfficientNetB3 (sinistra) e EfficientNetB5 (destra).

Dalla lettura preliminare delle matrici di confusione si osservano alcune tendenze interessanti. In particolare, **EfficientNetB3** ottiene il numero più elevato di veri positivi e il numero più basso di falsi negativi, suggerendo una maggiore sensibilità nella rilevazione delle piante malate. Tuttavia, questa performance è accompagnata da un incremento dei falsi positivi, indicando una certa tendenza alla sovrastima dei casi positivi.

Modelli come **ResNet50** e **InceptionV3** mostrano invece un comportamento più conservativo, con falsi positivi più contenuti ma una minore capacità di identificare tutti i positivi (più falsi negativi). Le due architetture **VGG** si collocano in una

posizione intermedia, offrendo un buon bilanciamento tra sensibilità e specificità, ma senza eccellere in modo netto.

Analisi dei risultati

Analogamente a quanto fatto nella fase sperimentale con immagini RGB, è stata condotta una valutazione dei modelli addestrati su immagini multispettrali considerando tre metriche principali: *accuracy*, *precision* e *recall*. Come in precedenza, sia la *precision* che il *recall* si riferiscono alla classe “malata”, ovvero ai casi positivi che rappresentano l’obiettivo principale del sistema di classificazione.

Tabella 4.2: Prestazioni dei modelli nella classificazione di immagini multispettrali. Precision e recall si riferiscono alla classe “malata”.

Modello	Accuracy (%)	Precision (%)	Recall (%)
VGG16	69.4	67.0	76.2
VGG19	70.6	70.4	70.7
ResNet50	65.7	68.0	59.0
InceptionV3	68.6	70.9	62.9
EfficientNetB3	68.8	65.7	78.5
EfficientNetB5	67.4	65.9	71.9

Le metriche ottenute nella fase multispettrale confermano in parte i risultati osservati nella sperimentazione RGB: la rete **VGG19** risulta nuovamente la più bilanciata in termini di accuratezza complessiva e precisione, ottenendo le migliori prestazioni in due su tre metriche. Tuttavia, si osserva come il modello **EfficientNetB3** raggiunga il **recall** più elevato, dimostrando una maggiore sensibilità nella rilevazione delle piante malate, aspetto particolarmente rilevante in scenari in cui la priorità è ridurre i falsi negativi.

Dal confronto tra le prestazioni ottenute con immagini RGB e quelle derivanti dall’utilizzo di immagini multispettrali emerge che i modelli, nel complesso, mostrano una maggiore stabilità nelle predizioni quando vengono alimentati con le bande multispettrali. Questo è evidente dal fatto che, in quest’ultima configurazione, le metriche di *accuracy*, *precision* e *recall* risultano più bilanciate tra loro, con una differenza massima tra il valore più alto e quello più basso che non supera i 13 punti percentuali per ciascun modello, mentre nella configurazione RGB tale differenza può arrivare fino a 37 punti percentuali. Tale comportamento suggerisce che le bande multispettrali contribuiscano a una rappresentazione più uniforme dell’informazione, riducendo l’asimmetria tra le metriche osservata in alcune configurazioni basate su immagini RGB.

Un altro aspetto rilevante è l'incremento dell'*accuracy* in quasi tutti i modelli analizzati, ad eccezione della rete **VGG19**. In quest'ultima, si osserva invece una leggera diminuzione omogenea in tutte e tre le metriche rispetto alla fase RGB, segno che l'introduzione delle bande multispettrali non ha portato a un vantaggio netto, ma neppure a un degrado severo delle prestazioni.

Nel complesso, le immagini multispettrali sembrano rendere le predizioni più coerenti e meno sbilanciate tra i vari modelli, pur non garantendo un miglioramento netto in ogni singolo caso.

Per rendere visivamente più immediato il confronto tra i modelli nelle due configurazioni, la Figura 4.24 riassume le performance in termini di *accuracy* e *F1-score*, suddivise per classe. In particolare, il *F1-score* rappresenta una media armonica tra *precision* e *recall*, ed è calcolato come:

$$F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (4.5)$$

Questa metrica risulta particolarmente utile in scenari sbilanciati, poiché penalizza le situazioni in cui una delle due metriche è significativamente inferiore all'altra, offrendo così una visione più equilibrata delle prestazioni del modello per ciascuna classe.

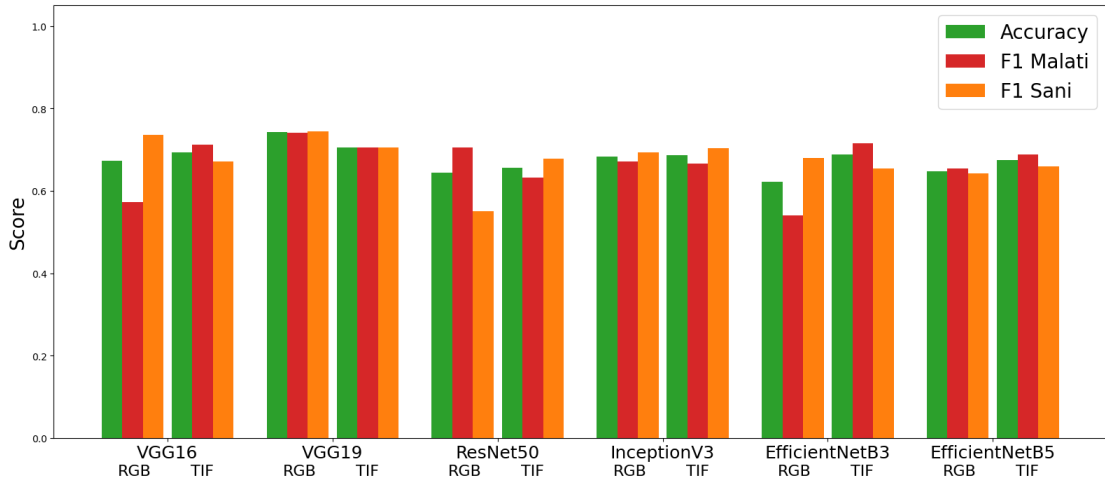


Figura 4.24: Confronto delle prestazioni tra modelli addestrati su immagini RGB e multispettrali in termini di *accuracy* e *F1-score* separato per le classi “malato” e “sano”. Si osserva una maggiore stabilità nelle predizioni per i modelli multispettrali, con valori più bilanciati tra le due classi.

Capitolo 5

Conclusioni e Sviluppi Futuri

I risultati ottenuti in questa tesi dimostrano che è possibile classificare in maniera semi-automatica lo stato di salute degli alberi di nocciolo con buoni livelli di accuratezza, utilizzando immagini UAV e modelli di deep learning. Sono state testate varie architetture, tra cui VGG16, VGG19, ResNet50, InceptionV3 ed EfficientNet, su immagini sia RGB che multispettrali.

Tra tutte le soluzioni esplorate, la rete **VGG19 addestrata su immagini RGB** ha mostrato le migliori prestazioni complessive con un'accuratezza del 74.3%. Questo modello si è distinto per la stabilità durante l'addestramento, per l'assenza di overfitting evidente e per un'elevata accuratezza di classificazione. Ha inoltre mostrato una curva di apprendimento regolare e una buona capacità di generalizzazione, raggiungendo risultati superiori rispetto alle reti concorrenti.

L'uso delle immagini multispettrali in formato .TIF ha prodotto risultati promettenti, con training accuracy spesso superiori all'80%, ma ha messo in evidenza la necessità di introdurre tecniche di regularization più sofisticate. Senza un'adeguata strategia di generalizzazione, infatti, il modello tende a sovra-adattarsi ai dati di addestramento, riducendo la sua efficacia su dati non visti. Si è osservato che il potenziale di tali immagini è elevato, ma per sfruttarlo appieno è necessario uno studio mirato su dropout, early stopping, weight decay e tecniche di data augmentation avanzate, calibrate in funzione della specifica natura delle bande multispettrali.

Un'altra criticità emersa riguarda la dimensione del dataset. I risultati ottenuti suggeriscono che un ampliamento consistente della base di immagini etichettate potrebbe permettere di migliorare ulteriormente le prestazioni, anche con architetture più complesse. L'incremento del numero di esemplari, soprattutto in condizioni di malattia iniziale o con variabilità ambientale maggiore, aiuterebbe i modelli a

generalizzare senza compromettere l'accuratezza.

In futuro, sarà pertanto utile combinare tre direzioni principali: un'estensione del dataset, una regolarizzazione più attenta dei modelli, e una valutazione più approfondita del potenziale offerto dalle immagini multispettrali. Sarà inoltre interessante sperimentare l'impiego e valutare approcci multimodali che combinino RGB e TIF in un'unica rete.

In conclusione, questa tesi ha mostrato che, con una pipeline di preprocessing accurata e un training ben strutturato, è possibile ottenere risultati robusti anche con sole immagini RGB, ma ha anche aperto la strada a ulteriori sviluppi per migliorare la precisione, l'affidabilità e l'applicabilità del sistema.

Bibliografia

- [1] M. Guijarro, G. Pajares, I. Riomoros, P.J. Herrera, X.P. Burgos-Artizzu e A. Ribeiro. «Automatic segmentation of relevant textures in agricultural images». In: *Computers and Electronics in Agriculture* 75.1 (2011), pp. 75–83. ISSN: 0168-1699. DOI: <https://doi.org/10.1016/j.compag.2010.09.013>. URL: <https://www.sciencedirect.com/science/article/pii/S0168169910001924> (cit. a p. 13).
- [2] Francesco Adamo, Filippo Attivissimo, Attilio Di Nisio, Mattia Alessandro Ragolia e Marco Scarpetta. «A New Processing Method to Segment Olive Trees and Detect Xylella Fastidiosa in UAVs Multispectral Images». In: *2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. 2021, pp. 1–6. DOI: [10.1109/I2MTC50364.2021.9459835](https://doi.org/10.1109/I2MTC50364.2021.9459835) (cit. a p. 14).
- [3] Yayong Chen et al. «Citrus Tree Segmentation from UAV Images Based on Monocular Machine Vision in a Natural Orchard Environment». In: *Sensors* 19.24 (2019). DOI: [10.3390/s19245558](https://doi.org/10.3390/s19245558). URL: <https://www.mdpi.com/1424-8220/19/24/5558> (cit. alle pp. 15, 21).
- [4] Google LLC. *Google Earth Pro*. <https://www.google.com/earth/> (cit. a p. 27).
- [5] DJI. *P4 Multispectral Specifications*. <https://www.dji.com/it/p4-multispectral/specs> (cit. a p. 27).
- [6] J. Lawless. *What Is A Neural Network?* <https://purplegriffon.com/blog/what-is-a-neural-network> (cit. a p. 31).
- [7] A. Elgendy. *What is Overfitting in Computer Vision? How to Prevent It*. <https://www.ultralytics.com/it/blog/what-is-overfitting-in-computer-vision-how-to-prevent-it> (cit. a p. 33).
- [8] Jacob Murel e Eda Kavlakoglu. *What is Data Augmentation?* <https://www.ibm.com/it-it/think/topics/data-augmentation> (cit. a p. 35).

- [9] Hasan Maher e Manar Kashmola. «Performance Improvement of Convolutional Neural Network Architectures for Skin Disease Detection». In: *International Journal of Computing and Digital Systems* 13 (ago. 2023), pp. 2210–142. DOI: 10.12785/ijcds/130152 (cit. a p. 42).
- [10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke e Andrew Rabinovich. «Going deeper with convolutions». In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9. DOI: 10.1109/CVPR.2015.7298594 (cit. a p. 44).
- [11] Umit Atila, Murat Ucar, Kemal Akyol e Emine Uçar. «Plant leaf disease classification using EfficientNet deep learning model». In: *Ecological Informatics* 61 (mar. 2021), p. 101182. DOI: 10.1016/j.ecoinf.2020.101182 (cit. a p. 46).
- [12] Mingxing Tan e Quoc V. Le. «EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks». In: *Proceedings of the 36th International Conference on Machine Learning*. 2019. URL: <https://arxiv.org/abs/1905.11946> (cit. a p. 46).