Weather Image Recognition

Francesca Motta (matr. 830107) - Davide Luperi (matr. 826249) - Maura Coniglione (matr. 874025)
Università degli Studi di Milano-Bicocca

Febbraio 2022

Il task di Image Recognition negli ultimi anni è stato reso possibile dall'avvento di modelli di Deep Learning sempre più robusti e in grado di garantire ottime performance.
L'obiettivo del lavoro è il riconoscimento di diversi tipi di condizioni climatiche sulla base di rilevazioni fotografiche.
L'analisi esplorativa ha evidenziato la presenza di situazioni climatiche molto simili, per esempio neve e brina, per cui si è considerato anche un dataset in cui si è fatto il merge delle classi in questione.
Si sono quindi implementati e valutati modelli di classificazione con 11 classi e 9 classi.
Dopo aver implementato i modelli basati sulle Convolutional Neural Networks, reti neurali adatte al trattamento di immagini, se ne è valutata la bontà in termini di accuracy.

1 Introduzione

Il lavoro si è concentrato sulla classificazione multiclasse di diversi tipi di condizioni metereologiche basandosi su rilevazioni fotografiche.
Il dataset a disposizione è costituito da 6862 immagini etichettate in 11 diverse condizioni climatiche. L'analisi esplorativa ha suggerito la presenza di due coppie di classi molto simili. Per esempio, molte delle immagini appartenenti alla classe 'rime' rappresentano panorami difficilmente distinguibili da paesaggi innevati che invece dovrebbero appartenere alla classe 'snow'. Si è affrontato il potenziale problema comparando la classificazione a 11 classi con una a 9, in cui due di queste sono costituite dal merge delle classi in cui si è riscontrato il problema in questione. Dalla comparazione è seguita la necessità di utilizzare una rete in grado di cogliere differenze più sottili tra le classi.

2 Materiali

Il dataset a disposizione è contenuto in Kaggle.[1]
Si tratta di un dataset contenente 6862 immagini riferite a 11 diversi tipi di condizioni climatiche.

La numerosità di ognuna delle 11 classi è la seguente:

Condizione metereologica	numerosità
Dew	698
Fogsmog	851
Frost	475
Glaze	639
Hail	591
Lightning	377
Rain	526
Rainbow	232
Rime	1160
Sandstorm	692
Snow	621

Tabella 1: Numerosità classi

Poichè il dataset originale non presentava lo split tra il training set, il validation e il test set, il primo problema affrontato è stato creare tali cartelle. Per farlo, è stato necessario rinominare le immagini con il formato 'NomeclasseNumero', dove Nomeclasse corrisponde al nome della classe di appartenenza dell'immagine e Numero alla posizione in cui si trova all'interno della cartella.

Inoltre, dall'analisi esplorativa si è osservato che ci sono coppie di cartelle contenenti immagini con caratteristiche pressochè identiche. Questo è un problema poichè la difficile distinzione potrebbe portare a un classificatore poco accurato.
Nello specifico, le immagini che si trovano nelle cartelle *Frost* e *Glaze* rappresentano entrambe paesaggi ghiacciati.

Figura 1: Frost

Figura 2: Glaze

Analogamente, le cartelle *Snow* e *Rime* contengono foto di paesaggi innevati in cui la brina non si distingue dalla neve.

Figura 3: Snow

Figura 4: Rime

Si ricordi che il dataset a disposizione non è di grande dimensione come sarebbe preferibile per l'utilizzo di architetture Deep. Con pochi dati, le neural networks possono infatti tendere a memorizzare il train, rientrando quindi nel problema di overfitting, oppure a non avere abbastanza dati per ottenere una buona stima dell'enorme quantità di parametri delle architetture Deep.
Per cui, con lo scopo di migliorare le performance dei classificatori, è stato costruito un dataset in cui le foto contenute nelle cartelle contenenti simili condizioni climatiche sono state spostate in un'unica cartella. Il dataset ottenuto è quindi caratterizzato da 9 cartelle tra cui la cartella *Frost and Glaze* e la cartella *Snow and Rime*. Avendo alleviato il problema di classificazione riducendolo a 9 classi e rimuovendo i casi di difficile discriminazione, l'aspettativa è di trovare classificatori più accurati.
Nel lavoro si sono quindi sviluppati un classificatore a 11 classi e uno a 9 classi. Per entrambi i dataset è stata effettuata la seguente divisione:

- 20% delle osservazioni di ciascuna classe è stato usato per creare il **Validation Set**
- 60% delle osservazioni di ciascuna classe è stato usato per creare il **Training Set**
- 20% delle osservazioni di ciascuna classe è stato usato per creare il **Test Set**

3 Metodi

I modelli implementati sono basati sulle Convolutional Neural Networks (CNN). L'idea alla base delle CNN è di applicare un filtro - o operatore di convoluzione - alle immagini in modo tale da evidenziarne i dettagli facilitando così la classificazione dell'immagine. La struttura generale delle reti CNN si basa sull'alternanza di **Layer convoluzionali** che, applicando filtri all'immagine, tramite delle trasformazioni ne prendono delle riduzioni, e **Layer di pooling** che hanno il compito di estrarre le caratteristiche locali delle immagini tramite sintesi statistiche. A tali strati seguono dei **Layer feed foward fully connected**, tipici delle reti neurali classiche, che sfociano poi negli strati di output, atti alla classificazione. Inoltre, il **Layer flatten**, situato prima del layer fully connected, è necessario per trasformare

il tensore 3D in uno 1D. In questo caso è stato utilizzato anche un **Layer di dropout**, volto a disattivare casualmente i nodi, e quindi le connessioni tra gli strati, in modo da ridurre la possibilità di overfitting.
Il lavoro si è sviluppato seguendo due approcci :

- CNN costruita ad hoc ed allenata sia sul dataset a 11 classi, sia su quello a 9. L'obiettivo di tale step è capire se la rete allenata su 9 classi ha davvero una capacità di classificazione migliore e quindi se la rete conferma l'ipotesi che l'analisi esplorativa iniziale aveva suggerito;
  Il modello, applicato ad entrambi i dataset, si compone di:
  - due layer convoluzionali aventi filtri di dimensione 30 e 16 e due di pooling (max pooling)
  - un tensore di input di dimensione (150, 150, 3)
  - un layer flatten
  - un layer di dropout con probabilità al 50
  - due layer densi fully connected, l'ultimo dei quali ha funzione di attivazione softmax in quanto ci troviamo nel caso di classificazione multiclasse.

  Sono quindi seguite le fasi di allenamento e valutazione della performance del modello per ciascun dataset.
  Per i risultati si veda la sezione successiva.

- Fine tuning: una tecnica che sfrutta reti pre-allenate e le concatena ad una rete fully connected. Infatti, una volta testata l'ipotesi al punto precedente, si è ritenuto di ripetere la classificazione a 11 classi con una rete più complessa, in grado di catturare differenze più sottili.
  Il fine tuning permette di scongelare una parte di una rete pre-esistente, che è stata in precendenza allenata su un dataset di grandi dimensioni, per poi applicarla al dataset d'interesse. Tale approccio si rivela utile in particolar modo nei casi in cui la dimensione del dataset è ridotto. Il modello preallenato scelto è VGG16 ed è contenuto all'interno del pacchetto Keras[2].
  In particolare, la rete presenta alcune peculiarità :
  - si è scelto un input shape di (450, 450) anzichè (150, 150) in quanto si è osservato su prove precedenti che tale formato di input permette un miglior risultato in termini di accuracy e loss, a costo di un dispendio computazionale notevole.
  - per ovviare a tale problema, si è preferito ridurre il numero di epoche da 100 a 10.
  - nell'algoritmo di ottimizzazione è stato utilizzato un learning rate particolarmente basso per ridurre al minimo le modifiche alla rete oggetti di fine tuning.

4 Risultati

L'evidenza empirica data dall'applicazione della rete neurale più semplice sul dataset iniziale e sul dataset modificato, depone a favore dell'ipotesi iniziale assunta: il classificatore opera meglio con 9 classi anzichè con 11.

In particolare, come è possibile notare dai grafici successivamente riportati, con 11 classi si ottiene il gomito dopo 25 epoche in corrispondenza di cui si raggiunge un'accuracy di circa **0.55** sia su validation che sul training e una loss di 1.4.
Con 9 classi il classificatore ha una variazione netta di pendenza poco dopo le 25 epoche in cui si registra un accuracy di **0.65** e una loss di 1.1.

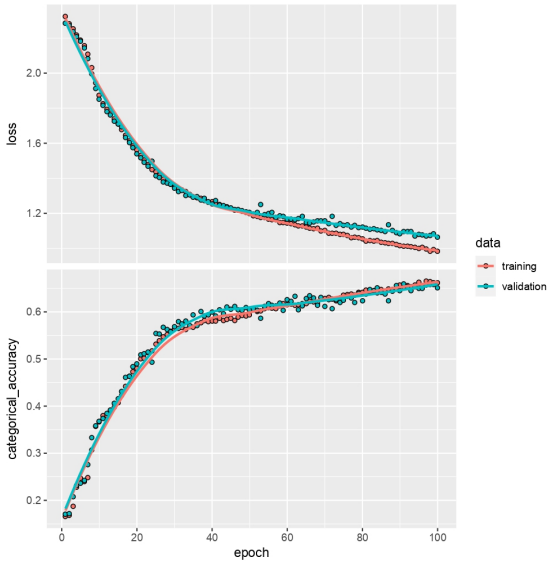

Figura 5: 11 classi

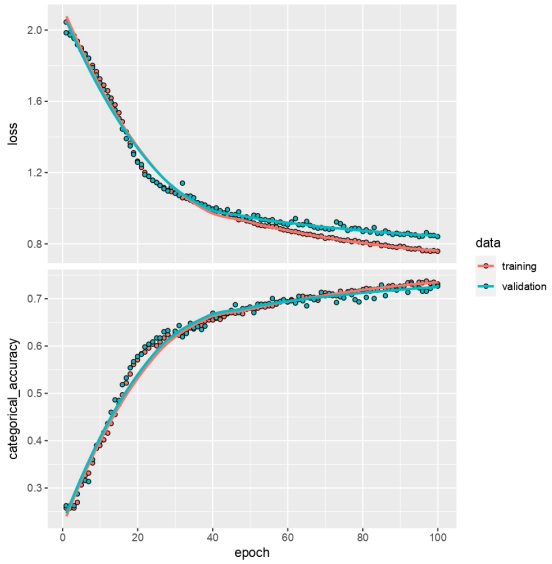

Figura 6: 9 classi

Passando all'approccio di fine tuning, la rete pre-allenata *VGG16* seguita da una rete fully connected raggiunge dei risultati molto soddisfacenti già sul dataset originario suddiviso in 11 classi.

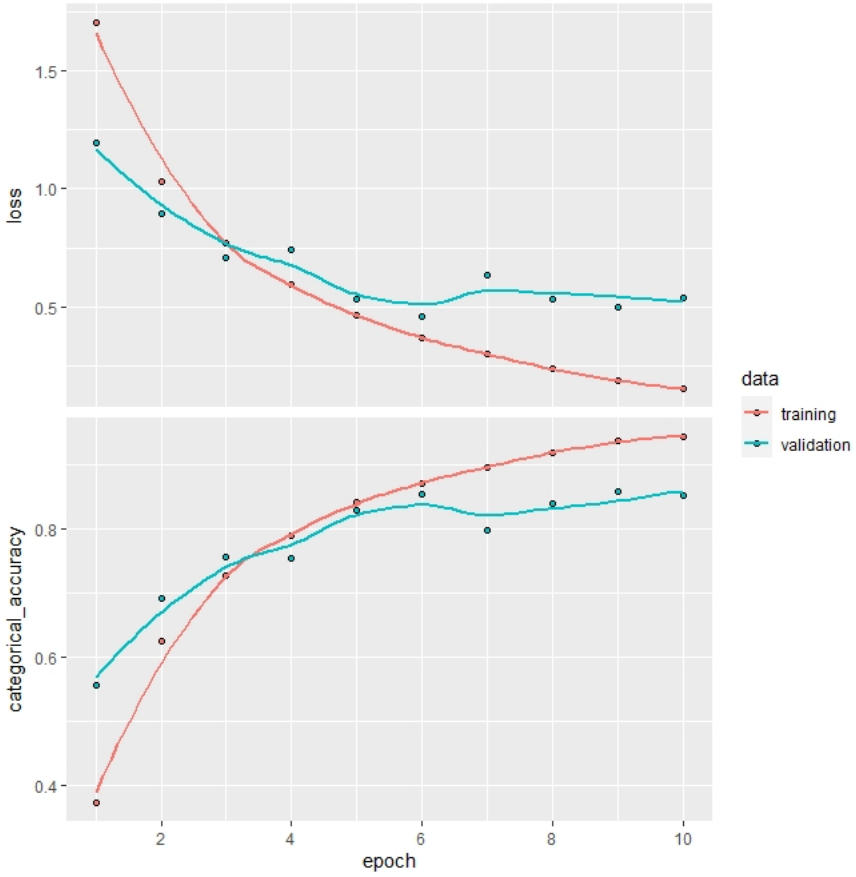

Figura 7: CNN fine tuning 11 classi

Nonostante si siano implementate solo 10 epoche, si nota che oltre alla quinta epoca i valori della loss e dell'accuracy nel validation non subiscono miglioramenti significativi. Al contrario, le stesse metriche sul training set continuano a migliorare. Perciò, per non incorrere in casi di overfitting, si sono salvati i parametri stimati alla quinta epoca.

- *Loss training* : 0.45
- *Accuracy training* : 0.84
- *Loss validation* : 0.55
- *Accuracy validation* : 0.83

Tale risultato, oltre che rappresentare la miglior performance ottenuta sul validation set, dati i limitati mezzi computazionali a disposizione, sottolinea che le reti convoluzionali sono in grado di captare differenze apparentemente indistinguibili anche all'occhio umano, con una buon grado di efficienza.

Alla luce di questi risultati, si è ristimata quest'ultima rete arrestata alla quinta epoca al fine di testarla sul test set. I risultati della valutazione sono riportati in seguito:

- *Loss* : 0.529
- *Accuracy* : 0.834

Per un confronto rapido si vedano in seguito le tabelle riassuntive con tutti i risultati ottenuti.

Metriche	valori
Accuracy 11 classi	0.55
Accuracy 9 classi	0.65
Loss 11 classi	1.4
Loss 9 classi	1.1

Tabella 2: Confronto modello semplice su 9 e 11 classi

Metriche	valori
Loss training	0.45
Accuracy training	0.84
Loss validation	0.55
Accuracy validation	0.83
Loss test	0.529
Accuracy test	0.834

Tabella 3: Risultato CNN fine tuning 11 classi

5 Conclusioni e sviluppi futuri

Il lavoro svolto ha raggiunto l'obiettivo prefissato di attuare un efficace classificazione di immagini mediante le reti neurali convoluzionali.
L'accuracy dell'83% ottenuta sul test set risulta molto soddisfacente alla luce del fatto che, come precedentemente anticipato, le reti neurali richiedono dataset di dimensioni molto più ampi rispetto a quello utilizzato per performare al meglio.
Inoltre, è opportuno sottolineare che un problema di classificazione multiclasse come quello affrontato risulta nettamente più complesso di un problema di classificaione binaria. Uno sviluppo futuro potrebbe essere quello di ridurre il problema a una classificazione one versus all.

I principali problemi riscontrati nel corso del lavoro sono stati l'utilizzo del software R[3], al quale è preferibile Python[4] per lo sviluppo in generale di reti neurali, e la potenza computazionale dei computer a nostra disposizione, spesso non sufficiente per elaborare i modelli creati in tempi ragionevoli.
I principali problemi riscontrati nel corso del lavoro sono stati i seguenti. L'utilizzo del software R[3] non è stata la scelta ottimale. Per lavori futuri è preferibile il sotware Python[4] per lo sviluppo di reti neurali e soprattutto per l'utilizzo della libreria Keras[2]. Inoltre, la potenza computazionale dei computer usati per questo progetto non è stata spesso sufficiente per stimare i modelli creati in tempi ragionevoli.

Bibliografia e risorse online

[1] *Weather Image Recognition Dataset*. 2021. URL: https://www.kaggle.com/jehanbhathena/weather-dataset.

[2] Francois Chollet et al. *Keras*. 2015. URL: https://github.com/fchollet/keras.

[3] RStudio Team. *RStudio: Integrated Development Environment for R*. RStudio, PBC. Boston, MA, 2020. URL: http://www.rstudio.com/.

[4] Guido Van Rossum e Fred L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.