

Advanced Machine Learning: Assignment #4

Fabrizio D'Intinosante — 838866

Università degli Studi di Milano Bicocca — November 21, 2019

Introduzione

L'obiettivo previsto per questo *assignment* è quello di effettuare *transfer learning* utilizzando una *Convolutional Neural Network* pre-addestrata sul noto *dataset IMAGENET*. Il task scelto è la classificazione di immagini raffiguranti 5 classi di fiori: rose, denti di leone, margherite, girasoli e tulipani. Il *dataset* è reperibile su kaggle al seguente [link](#). Come rete pre-addestrata si è scelta una VGG16 mentre, come modello di classificazione, dopo diverse prove, si è scelto di operare attraverso una *Support Vector Machine*.

1 Esplorazione e preprocessing

Il *dataset* scelto è composto da 4323 immagini 320x240 con canali RGB. Queste immagini, come detto, rappresentano 5 classi diverse di fiori, ritratte in situazioni molto differenti: fotografate in gruppo, singolarmente, mentre sono tenute in mano da una persona oppure all'interno di un vaso; è quindi quasi sempre presente un altro elemento all'interno dell'immagine come è possibile vedere dall'esempio in fig. 1. La distribuzione delle classi all'interno del *dataset* è visualizzabile in fig. 2.

In origine le immagini si presentano non etichettate ma, piuttosto, contenute per ogni classe all'interno di una cartella nominata con il nome dell'etichetta a loro assegnata. Per questa ragione si è reso necessario realizzare una breve funzione per importare i nostri dati etichettandoli con il nome delle cartelle in cui erano contenuti; nel fare questo, inoltre, si è scelto arbitrariamente di ritagliare le immagini in un formato 224x224 mantenendo i canali RGB.

Per pre-processare i dati si è scelto di ricorrere al medesimo *preprocessing* applicato alle immagini del *dataset IMAGENET*, utilizzate per addestrare la rete VGG16 importata.

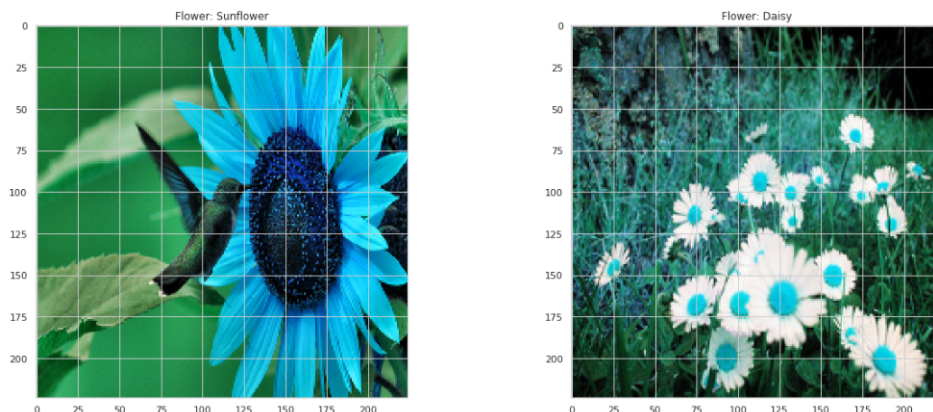


Figure 1: Esempio di immagini

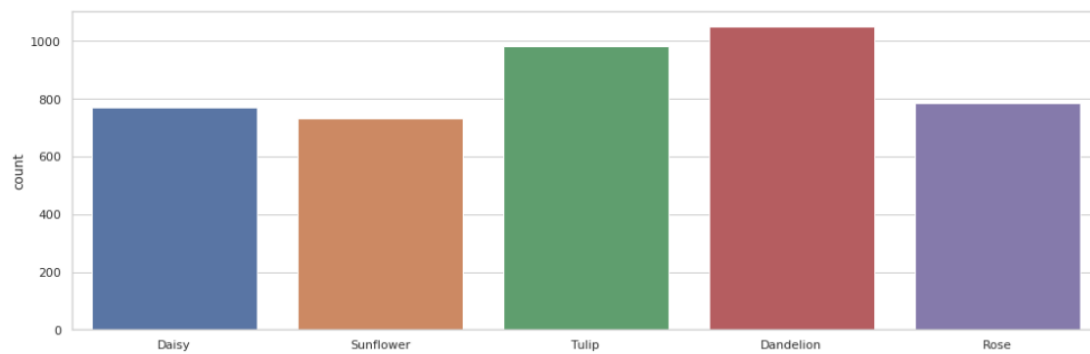


Figure 2: Distribuzione delle classi dei fiori

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590880
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590880
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0

Figure 3: Schema di taglio dei layers di VGG16 per i diversi modelli

2 Modelli

Per assolvere all'obiettivo posto dall'*assignment* si è deciso di creare sostanzialmente 3 modelli basati sulle *features* estratte da 3 diversi layers della rete:

- il primo taglio è avvenuto all'altezza del layer **block5_pool**;
- il secondo al layer **block4_pool**;
- l'ultimo al layer **block3_pool**.

Uno schema esemplificativo dei tagli effettuati è visibile in fig. 3.

Per ogni diverso taglio del modello pre-addestrato si è proceduto sostanzialmente ad estrarre le *features* prodotte dal passaggio delle immagini nella rete: nell'ordine **25088**, **100352** e **200704**. Successivamente si è proceduto ad effettuare una PCA in modo da conservare l'80% della varianza spiegata, con l'obiettivo di ridurre la dimensionalità delle *features*, così da permettere alla *Support Vector Machine* di operare in tempi accettabili e con *performances* comunque buone.



NB Prima di essere sottoposte alla PCA, le *features* sono state scalate in un range tra 0 e 1, così da renderle compatibili con il modello SVM.

I 3 modelli di classificazione realizzati sono stati ottimizzati nei loro iper-parametri attraverso un meccanismo di *grid search*, così da massimizzarne le prestazioni:

- C: [5, 10, 15];

```

Best C value: 5
Best gamma value:auto

              precision    recall  f1-score   support

     0       0.835        0.844        0.839        192
     1       0.915        0.905        0.910        263
     2       0.787        0.888        0.835        196
     3       0.853        0.755        0.801        184
     4       0.881        0.870        0.875        246

 accuracy          0.858        1081
 macro avg         0.854        0.852        0.852        1081
 weighted avg      0.859        0.858        0.857        1081

```

Figure 4: Prestazioni modello con *features* estratte da block5_pool su *test set*

```

Best C value: 5
Best gamma value:auto

              precision    recall  f1-score   support

     0       0.778        0.823        0.800        192
     1       0.877        0.871        0.874        263
     2       0.798        0.806        0.802        196
     3       0.818        0.804        0.811        184
     4       0.866        0.837        0.851        246

 accuracy          0.832        1081
 macro avg         0.827        0.828        0.828        1081
 weighted avg      0.833        0.832        0.832        1081

```

Figure 5: Prestazioni modello con *features* estratte da block4_pool su *test set*

- kernel: [rbf, linear, poly, sigmoid];
- gamma: [auto, scale].



NB Prima di procedere al *training* del modello SVM si è proceduto a suddividere le istanze in *training set* e *test set* in proporzioni 75% e 25%.

Il primo modello realizzato, che è anche il migliore in quanto a prestazioni sul *test set* come è possibile vedere in fig. 4, è quello costruito partendo dalle *features* estratte dal layer **block5_pool**, ovvero quello più vicino alla coda della rete tra quelli presi in esame. La PCA in questo caso, per conservare l'80% della varianza spiegata, ha estratto circa 1500 componenti. I parametri ottimi trovati sono $C = 5$, kernel = rbf e gamma = auto.

Il secondo modello, realizzato con l'obiettivo di risalire la rete per verificare l'eventuale impatto positivo di *features* più generiche estratte dalle immagini ha prodotto, al contrario delle aspettative, delle prestazioni leggermente inferiori rispetto al modello precedente, come è possibile osservare in fig. 5. La PCA nel caso di questo modello, a parità di quota di varianza spiegata, ha estratto circa 2000 componenti, mentre la *grid search* ha prodotto come parametri ottimali anche per questo $C = 5$, kernel = rbf e gamma = auto.

```

Best C value: 10
Best gamma value:scale

              precision    recall  f1-score   support

     0       0.756        0.677        0.714        192
     1       0.841        0.802        0.821        263
     2       0.783        0.719        0.750        196
     3       0.750        0.815        0.781        184
     4       0.727        0.821        0.771        246

 accuracy          0.772        1081
 macro avg         0.771        0.767        0.768        1081
 weighted avg      0.774        0.772        0.771        1081

```

Figure 6: Prestazioni modello con *features* estratte da block3_pool su *test set*

Il terzo ed ultimo modello infine è stato realizzato utilizzando le *features* estratte ad un livello ancora più elevato della rete, il **block3_pool** e tra quelli realizzati è quello che ha raggiunto le prestazioni più modeste, come visibile in fig. 6. Anche per questo modello la PCA ha estratto circa 2000 componenti e come parametri ottimali sono stati scelti $C = 10$, $\text{kernel} = \text{rbf}$ e $\text{gamma} = \text{scale}$.

3 Conclusioni

In conclusione, si può affermare che il modello che ha prodotto le migliori *performances* sul *test set* sia il primo i cui risultati sono riportati in fig. 4. Sembrerebbe infatti che le *features* estratte sostanzialmente in concomitanza del finale della rete risultino significativamente più esplicative rispetto a quelle più generiche prodotte nella parte iniziale della VGG16. Ciò potrebbe essere dovuto al fatto che all'interno del *dataset* **IMAGENET** sono presenti, tra le tantissime immagini, anche molte rappresentanti fiori e che, per questo, il modello risulti produrre delle *features* di maggiore qualità con l'avvicinarsi dei *layers* alla coda della rete.