Riconoscimento di Attività Umane con Perceptron
Intelligenza Artificiale

Lapo Bartolacci

January 13, 2020

1 Introduzione

I moderni smartphone hanno la possibilità di sfruttare i molti sensori che hanno integrati. Dato il ruolo sempre più centrale di questi dispositivi nella vita di tutti i giorni, si prevede che questi dispositivi terranno continuamente traccia delle nostre attività, imparando da esse per aiutarci a prendere decisioni migliori. Il Riconoscimento di Attività ha lo scopo di identificare le azioni eseguite da un essere umano date delle informazioni percepite da esso o dall'ambiente. In questo elaborato vengono utilizzate le informazioni ricavate dai sensori presenti negli smartphone per classificare un insieme di attività (standing, walking, laying, walking, walking upstairs e walking downstairs) utilizzando l'algoritmo di apprendimento supervisionato Perceptron.

2 Metodo

2.1 Data Set

Il data set utilizzato è disponibile qui: `http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones`
I dati sono stati raccolti con un gruppo di 30 volontari di età compresa tra i 19 e i 48 anni. Ogni persona ha eseguito le 6 attività con uno smatphone alla vita, registrando i dati del giroscopio e dell'accelerometro ad una frequenza di 50 Hz. I dati sono stati partizionati in modo casuale in due insiemi, il 70% è stato utilizzato per il training e il 30% per il test.
Ulteriori informazioni su come sono stati processati i dati sono disponibili nella pagina del data set.

2.2 Implementazione

Il linguaggio di programmazione utilizzato è il Python, in particolare è stata utilizzata l'implementazione dell'algoritmo Perceptron fornita dalla libreria `Scikit-learn`. Per rendere Perceptron più performante, inizialmente viene fatto il feature scaling con il metodo di standardizzazione tramite il trasformatore `StandardScaler` .

```
sc = StandardScaler()
sc.fit(features_train)
features_train_std = sc.transform(features_train)
features_test_std = sc.transform(features_test)
```

Dopo aver creato un oggetto Perceptron, `GridSearchCV` si occupa di cercare la combinazione di parametri migliori per il classificatore. La valutazione viene fatta con *K-fold cross-validation* e come funzione di utilità viene utilizzato *F1 score* che combina precision e recall in un unico parametro.

```
ppn = Perceptron()
grid_search = GridSearchCV(ppn, param_grid, cv=5,
                  return_train_score=True, n_jobs=-1,
                  scoring='f1_weighted')
```

La funzione *fit* esegue il training con 7352 istanze ognuna con 561 features.

```
grid_search.fit(features_train, target_train)
```

La funzione *predict* viene utilizzata per testare il perceptron appena allenato con 2947 istanze di test.

```
bestPred = grid_search.predict(features_test)
```

3 Risultati

I risultati della classificazione di Perceptron per i dati di test sono riportati tramite matrice di confusione nella Table 1, dove sono anche mostrati l'accuratezza , la precisione e il recall. Sono state valutate 2947 istanze di test con circa lo stesso numero di istanze per classe. La maggior parte delle predizioni false si hanno per la classe sitting a differenza dei risultati riportati da Anguita et al. [1] nella Figure 1, dove le attività statiche hanno performato meglio rispetto a quelle dinamiche.

		Predette						
		Walking	Upstairs	Downstairs	Standing	Sitting	Laying	Recall %
Effettive	Walking	**488**	1	5	2	0	0	98.39
	Upstairs	9	**454**	7	1	0	0	96.39
	Downstairs	2	3	**411**	4	0	0	97.86
	Standing	0	0	0	**524**	8	0	98.5
	Sitting	0	2	0	85	**404**	0	82.28
	Laying	0	0	0	23	0	**514**	95.72
	Precision %	97.8	98.7	97.16	82.0	98.06	100.0	**94.84**

Table 1: Matrice di Confusione. Le righe rappresentano le classi effettive e le colonne le classi predette. La diagonale mostra le istanze classificate correttamente.

Method	MC-SVM							MC-HF-SVM $k = 8$ bits						
Activity	Walking	Upstairs	Downstairs	Standing	Sitting	Laying	*Recall %*	Walking	Upstairs	Downstairs	Standing	Sitting	Laying	*Recall %*
Walking	**109**	0	5	0	0	0	*95.6*	**109**	2	3	0	0	0	*95.6*
Upstairs	1	**95**	40	0	0	0	*69.8*	1	**98**	37	0	0	0	*72.1*
Downstairs	15	9	**119**	0	0	0	*83.2*	15	14	**114**	0	0	0	*79.7*
Standing	0	5	0	**132**	5	0	*93.0*	0	5	0	**131**	6	0	*92.2*
Sitting	0	0	0	4	**108**	0	*96.4*	0	1	0	3	**108**	0	*96.4*
Laying	0	0	0	0	0	**142**	*100*	0	0	0	0	0	**142**	*100*
Precision %	*87.2*	*87.2*	*72.6*	*97.1*	*95.6*	*100*	***89.3***	*87.2*	*81.7*	*74.0*	*97.8*	*94.7*	*100*	***89.0***

Figure 1: Matrice di confusione dei risultati riportati da Anguita et al. [1] utilizzando due versioni di un classificatore diverso.

References

[1] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *International workshop on ambient assisted living*, pages 216–223. Springer, 2012.