Reproduction of Multiple Instance Dictionary Learning for Activity Representation study

Visione artificiale e riconoscimento 2022/2023

Christian D'Errico

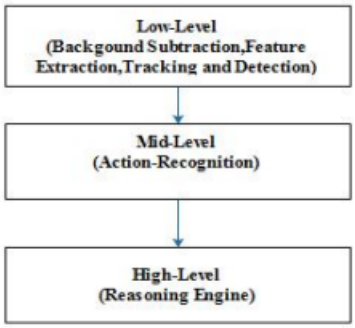
21-09-2023

1 HAR: Human activity recognition

Identificare le azioni che vengono compiute in un video automaticamente e classificarle è la funzionalità principe dei sistemi video più avanzati.
Lo scopo dell'analisi è identificare le attività e gli obiettivi di uno o più soggetti, a partire da una serie di osservazioni delle loro mosse e delle condizioni ambientali in cui queste vengono messe in atto: si tratta di un compito reso impegnativo da tutta una serie di ostacoli, come disturbi prodotti dallo sfondo, occlusione parziale degli elementi principali, cambiamenti di scala, punti di vista differenti e problemi di luminosità per esempio.
Molte applicazioni richiedono un sistema di riconoscimento di attività e si è pertanto creata una vera e propria disciplina che si occupa di tutto questo, disciplina che prende il nome di Human Activity Recognition (HAR).

2 Introduzione

Data l'importanza crescente dell'analisi dei video (visti i numerosi impieghi in ambiti che spaziano da scenari relativi a sicurezza e videosorveglianza, fino ad arrivare alla sfera videoludica e all'industria dei videogames), numerose sono state le soluzioni che si sono studiate per il problema.
Convenzionalmente però, quasi tutti i sistemi HAR presentano tale struttura gerarchica:



Al livello più basso troviamo background subtraction, feature extraction, tracking e detection: queste tecniche costituiscono il primo step della gerarchia e sono performate su pixel, blocchi o combinazioni di entrambi questi elementi dei frame.
È a questo livello che entrano in gioco modellazioni gaussiane, mixture di gaussiane, ViBe, filtri di Kalman, modelli Hidden Markov e altre tecniche pixel-based, mentre per modelli block-based, Normalized Distance Vector, similarità fra istogrammi, PCA incrementali, Local Binary Patterns e simili.
Il secondo step è costituito dal processo di riconoscimento delle azioni: un modello neurale, tipicamente, ha il compito di identificare specifiche azioni o com-

portamenti, basandosi sui dati e distinguendo fra diverse categorie di attività umane.
Infine, l'ultimo passo riguarda l'utilizzo di un motore di reasoning: un sistema di interpretazione avanzato, ad un livello di astrazione più elevato, cerca di analizzare le attività umane in un contesto ampio, tentando di comprendere il significato o l'intento che vi si cela dietro (ad esempio, il motivo o la finalità per cui una persona, filmata mentre corre, lo stia facendo).
Esistono due approcci che si distinguono per il tipo di analisi che viene effettuata sulle feature raccolte: il primo raccoglie tecniche model-based, che lavorano sulla costruzione di un modello umano cinematico per analizzare i movimenti e i gesti, il secondo riguarda tecniche feature-based (come quella oggetto di questo lavoro), che fondano appunto il loro successo sull'individuazione, estrazione e analisi di keypoint e descrittori locali.

2.1 Alcuni approcci

Fra i metodi feature-based, è importante citare vari contributi: il lavoro di H. Zhang and O. Yoshie, *Improving human activity recognition using subspace clustering*, che propone un metodo basato su clustering subspaziale, SCAR, per riconoscere e individuare attività anomale; *A general method for human activity recognition in video*, di Neil Robertson e I. Reid, che rappresenta le azioni come vettori di feature con informazioni sulla traiettoria (posizione e velocità), e un set di descrittori locali di movimento; *"Explicit modeling of human object interactions in realistic videos* di Prest et al., che hanno studiato una metodologia che richiede estrazione dei frame e applicazione di detector di oggetti o soggetti umani per l'identificazione; *A similarity measure for analyzing human activities using human-object interaction context* di Mohsen et al., che utilizzano una funzione kernel per calcolare la similarità fra due video basandosi sulla similarità fra le interazioni fra parti del corpo umane e oggetti; *Observing human object interactions: Using spatial and functional compatibility for recognition* di Gupta e al., che hanno elaborato un approccio che utilizza motion features e analizza il movimento della mano per raggiungere gli oggetti.

2.2 Metodo proposto

L'elaborato realizzato ripropone l'esperimento condotto nel paper *Multiple Instance Dictionary Learning for Activity Representation* (di Sabanadesan Umakanthan, Simon Denman, Clinton Fookes and Sridha Sridharan), focalizzato sulla realizzazione di un framework basato su un metodo bag-of-words.
Parecchi approcci HAR si basano sul concetto di bag-of-features (l'idea di rappresentare un video o un'immagine tramite un istogramma di occorrenze di alcune visual words che codificano specifiche caratteristiche locali) e cercano di superarne i limiti: X. Wang, B. Wang, X. Bai, W. Liu, and Z. Tu con *Max-margin multiple instance dictionary learning*, propongono una strategia per la creazione dei dizionari che impiega K-means come algoritmo di clustering per

l'individuazione delle diverse dimensioni dei codebook seguito dall'applicazione di un SVM multi-classe, che massimizza il margine di separazione fra i differenti clusters individuati; X. Zhang, H. Zhang, and X. Cao in *Action recognition based on spatial temporal pyramid sparse coding* usano Sparse Coding per quantizzare le caratteristiche e una piramide spazio-temporale per rappresentare un'azione, mentre in *Nonlinear learning using local coordinate coding* di K. Yu, T. Zhang, and Y. Gong viene proposta una tecnica di Local Coordinate Coding (LCC), un metodo che utilizza vincoli di località per proiettare ogni descrittore nel suo sistema di coordinate, integrato poi da max pooling per generare la rappresentazione finale.
Il metodo analizzato fa tesoro di tutti questi spunti e cerca di integrare nella maniera migliore possibile queste idee per realizzare un'architettura che risulti accurato a tal punto da superare le tecniche allo stato dell'arte: in primis, un modello SVM multiclasse è utilizzato per individuare le feature correlate ad ogni categoria di azione e poi impiegato un algoritmo di clustering (k-means) per generare un codebook per ogni classe di attività.
In seguito, si impiega una tecnica locality-constrained per la quantizzazione dei video, seguita da un pooling piramidale spazio-temporale per ottenere il filtraggio delle caratteristiche includendo anche informazioni di tipo spazio-temporale.

Infine, viene addestrato un ulteriore SVM multi-classe per classificare i video.

3 Fase 1: feature extraction

La fase iniziale prevede l'estrazione delle feature: si utilizza un estrattore ad-hoc proposto da H. Wang, M. M. Ullah, A. Klaser, I. Laptev, C. Schmid e documentato nel paper *On Space-Time Interest Points*.

3.1 Detector

Il detector è un'estensione spazio-temporale dell'Harris detector, ribatezzata Harris3D.

Viene calcolata una matrice spazio-temporale ad ogni punto nel video:

$$\mu(\cdot;\sigma,\tau) = G(\cdot;s\sigma,s\tau) * \begin{pmatrix} L_x^2 & L_xL_y & L_xL_t \\ L_xL_y & L_y^2 & L_yL_t \\ L_xL_t & L_yL_t & L_t^2 \end{pmatrix}$$

usando valori di scala spaziale e temporale indipendenti σ e τ, una funzione di smoothing gaussiana G e un parametro s che lega la scala di integrazione per G alle scale locali σ e τ.

Le derivate del primo ordine di una sequenza video v sono definite come:

$$L_x(\cdot;\sigma,\tau) = \frac{\partial}{\partial x}(G * v) \tag{1}$$

$$L_y(\cdot;\sigma,\tau) = \frac{\partial}{\partial y}(G * v) \tag{2}$$

$$L_t(\cdot;\sigma,\tau) = \frac{\partial}{\partial t}(G * v) \tag{3}$$

Le locazioni finali dei punti di interesse sono date dai massimi locali di:

$$H = \det(\mu) - k\,\mathrm{trace}(\mu), \quad H > 0. \tag{4}$$

3.2 HOG/HOF feature

Per caratterizzare il movimento e l'aspetto locale, ai keypoint vengono associate feature che sono il risultato di una fusione fra istogrammi delle orientazioni del gradiente (HOG) e del flusso ottico (HOF).
Gli istogrammi vengono calcolati o creati raccogliendo dati relativi alla distribuzione di pixel o caratteristiche all'interno dell'area spazio-temporale circostante i punti di interesse individuati, regione del descrittore che è data da un cuboide di dimensione $\Delta x(\sigma) = \Delta y(\sigma) = 18\sigma$ e $\Delta t(\tau) = 8\tau$.
Ogni cuboide è diviso in una griglia di celle $n_x \times n_y \times n_t$; per ogni cella, 4 bin sono assegnati all'istogramma HOG e 5 all'HOF (4 direzioni e un bin aggiuntivo) e le celle dei due istogrammi normalizzate sono poi concatenate. L'estrattore è stato applicato con i parametri di default.

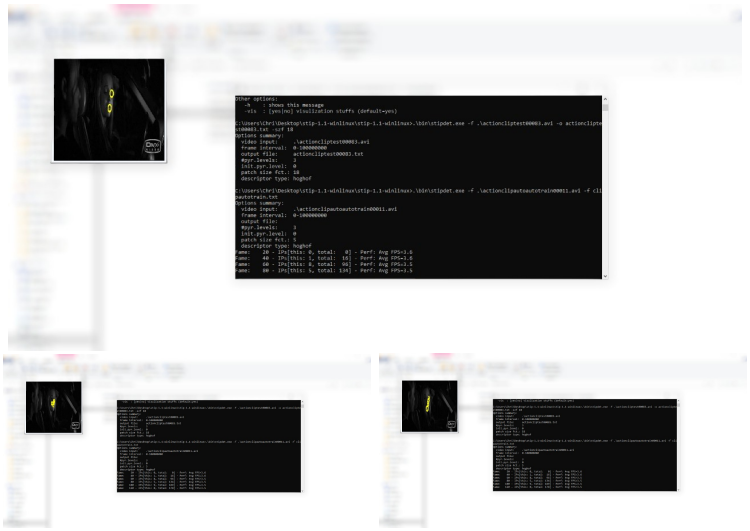


Figure 1: Esempio di utilizzo dell'estrattore SIPT

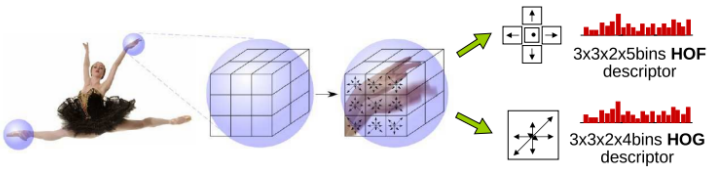


Figure 2: Un punto di interesse è descritto da un cuboide tridimensionale diviso in una griglia di celle; per ogni cella, un istogramma HOG, così come un HOF, sono concatenati

4 Multiple Instance Dictionary Learning

La fase di realizzazione dei codebook è MIL-based (Multiple Instance Learning).

Nel Multi-Instance Learning (MIL) viene dato un insieme di borse, indicato come $X = \{X_1, ..., X_n\}$. Ogni bag contiene un insieme di istanze, rappresentate come $X_i = \{x_{i1}, ..., x_{im}\}$, e ogni istanza è un vettore d-dimensionale $x_{ij} \in \mathbb{R}^{d\times 1}$. Inoltre, ogni bag è associata a un'etichetta di borsa $Y_i \in \{0, 1\}$, e ogni istanza è associata anche a un'etichetta di istanza $y_{ij} \in \{0, 1\}$. Il rapporto tra l'etichetta di una bag e le etichette di istanza è interpretato come segue: se $Y_i = 0$, allora $y_{ij} = 0$ per tutti $i, j \in [1, ..., m]$, il che significa che nessuna istanza nella bag è positiva. Se $Y_i = 1$, allora almeno un'istanza $x_{ij} \in X_i$ è positiva.
In partenza, non è dato sapere quale feature associata ad una bag positiva si possa considerare realmente positiva e quale no: si addestra un modello mi-SVM (un'estensione MI di un modello SVM) a riconoscere e distinguere feature positive da feature invece negative.
In questo caso, ogni video è pensato come una bag e, per ognuna delle k classi di attività, ne viene fissata una come positiva e le altre $k - 1$ come negative.
Si mostrano dunque i diversi step dell'algoritmo studiato per risolvere il problema MIL associato allo specifico contesto che è stato presentato: in primis, si seleziona un training set composto dalle feature più positive e da tutte quelle negative e, su questo, si addestra un modello SVM; il modello così ottenuto è in grado di predire le etichette delle feature (alla prima iterazione, video della classe positiva conterranno solamente feature positive, tutte le altre saranno negative) e di estrarne la "positività", in termini probabilistici, utilizzando la funzione softmax: questa indicazione servirà all'iterazione successiva dell'algoritmo per il campionamento del training set.
Prima di procedere oltre, la predizione viene però "corretta" nel rispetto dei vincoli del problema: se vi sono feature che dovrebbero essere negative queste vengono forzate ad essere negative, mentre, qualora il modello avesse predetto come negative tutte le feature di una bag positiva, quella a cui è associato un grado di positività maggiore viene etichettata come positiva.
Il criterio di arresto dell'algoritmo è stabilito sulla base di un numero massimo di iterazioni da svolgere, ma tutto potrebbe terminare anche qualora per nessuna feature venga più aggiornata l'etichetta di classe (viene raggiunta una situazione

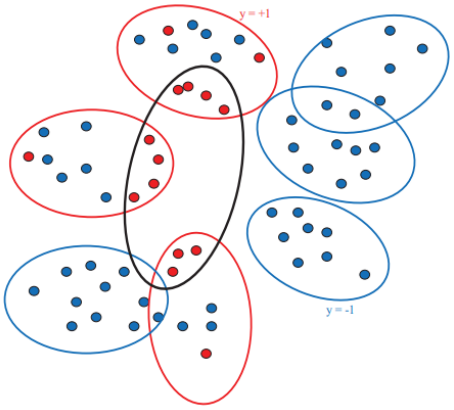


Figure 3: Illustrazione di mi-SVM per separare le istanze nei sacchetti positivi. Un video (bag) è rappresentato come una raccolta di caratteristiche (istanze), la bag viene etichettata come positiva se almeno una delle istanze (rosse) nella bag è positiva e questa viene considerata negativa se tutte le istanze (blu) sono negative. mi-SVM mira a trovare le istanze positive nelle bag positive massimizzando la separazione tra le istanze positive e negative (l'ellisse nera indica le istanze identificate come positive da mi-SVM).

di convergenza).
Al termine delle operazioni, il modello estratto (che massimizza il margine fra le due classi all'interno delle bag positive) viene utilizzato per predire le feature positive da utilizzare per costruire il codebook, per il quale si ricorre all'applicazione dell'algoritmo di clustering non supervisionato K-Means: a differenza di un approccio classico di BoW, non viene realizzato un unico dizionario rappresentativo di tutte le differenti classi, ma ne viene concepito uno per ogni classe, in modo che questi siano più specificatamente legati ad ogni singola categoria di azioni.

5 Fase 3: LLC Encoding

Nella fase di codifica delle caratteristiche, i descrittori, estratti dai video, vengono quantizzati usando i codebook.

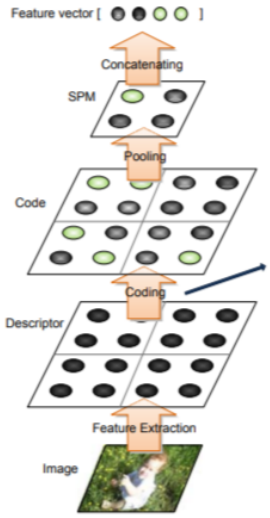


Figure 4: Confronto fra VQ, SC e LLC

Sebbene vi siano parecchi metodi per effettuare questa operazione, la Vector Quantization (VQ) è l'approccio più utilizzato.
VQ risolve questa equazione:

$$\arg\min_{C} \sum_{i=1}^{N} ||x_i - Bc_i||^2$$
$$\text{s.t. } ||c_i||_{\ell_0} = 1, \quad ||c_i||_{\ell_1} = 1, \quad c_i \geq 0, \quad \forall i$$

Dove $C = [c_1, c_2, ..., c_N]$ è il set di codici per X, ovvero la rappresentazione quantizzata del video.
Concretamente, l'obiettivo è trovare i coefficienti c_i che minimizzano la distanza fra il vettore di feature x_i, originariamente estratte, e il codebook, coefficienti che vengono utilizzati per scrivere come combinazione lineare degli stessi le feature.

Il vincolo $||c_i||_{\ell_0} = 1$ implica che vi sia soltanto un elemento in c_i diverso da 0, corrispondente all'id di quantizzazione di x_i, mentre il vincolo $||c_i||_{\ell_1} = 1, \quad c_i \geq 0$ fa in modo che il coefficiente lineare c_i associato alla caratteristica x_i sia non negativo (cioè maggiore o uguale a zero) e sia anche uguale a 1. Nella pratica significa che la caratteristica x_i viene codificata usando solo una delle feature del

codebook B (quella corrispondente all'indice indicato dal coefficiente non nullo) e questa feature ha un peso di 1 nella codifica. Quindi, per ogni caratteristica x_i nel set X, la codifica VQ, con i vincoli sopra descritti, seleziona esattamente una feature dal codebook B (indicata dall'indice del coefficiente non nullo), assegna a questa feature un peso di 1 e tutti gli altri pesi saranno nulli.
Questo è ciò che rende la codifica sparsa e causa perdita di informazione: per ovviare a questa problematica, si può rilassare il primo dei due vincoli sopra citati, usando un termine di regolarizzazione della sparsità; questo è cioè che fa la Sparse Coding:

$$\arg\min_{C} \sum_{i=1}^{N} \|x_i - Bc_i\|_2^2 + \lambda \|c_i\|_1$$

L'inserimento di un simile termine comporta ulteriori benefici:

- **Sovradimensionamento del codebook**: il codebook B, che contiene le feature utilizzate per rappresentare le caratteristiche, è di solito più grande della dimensione D delle caratteristiche stesse (ovvero $M > D$). Questo significa che il sistema è sotto-determinato, il che comporterebbe una molteplicità di soluzioni possibili. Per garantire una soluzione unica, è necessario utilizzare la regolarizzazione ℓ_1 (o sparsità) che porta a una rappresentazione con meno elementi non nulli nei vettori di codifica, che risolve il problema dell'indeterminazione e permette di ottenere una soluzione unica.
- **Cattura di schemi salienti**: la codifica sparsa fa in modo che la rappresentazione appresa sia focalizzata su schemi salienti all'interno delle caratteristiche locali. Ciò significa che si selezionano determinate feature significative e ne vengono ignorate altre meno rilevanti, migliorando così la rappresentazione delle caratteristiche.
- **Riduzione dell'errore di quantizzazione**: La codifica sparsa (sparse coding) può ottenere un errore di quantizzazione molto inferiore rispetto all'approccio tradizionale di quantizzazione vettoriale (VQ).

Tuttavia, la caratteristica di Località è più essenziale del concetto di Sparsità, poiché la località conduce alla sparsità e non vice-versa: da qui la scelta di utilizzare **LLC**.
Il principio di località in LLC, infatti, enfatizza la vicinanza spaziale o la similarità tra le caratteristiche o i dati in una determinata area o regione. Questo significa che, quando si codificano le caratteristiche, si cerca di rappresentare ciascuna caratteristica in un "sistema di coordinate locale" che tenga conto delle caratteristiche vicine o simili nello spazio.

LLC segue questi criteri:

$$\min_{C} \sum_{i=1}^{N} \|x_i - Bc_i\|_2^2 + \lambda \|d_i \odot c_i\|_2^2 \quad \text{s.t.} \quad \mathbf{1}^T c_i = 1, \quad \forall i$$

Dove $\odot$ indica la moltiplicazione elemento per elemento, e $d_i \in \mathbb{R}^M$ è l'adattatore di località che conferisce una diversa importanza alle basi del codebook a seconda della somiglianza con il descrittore di input x_i. In particolare,

$$d_i = \exp\left(\sigma \cdot \text{dist}(x_i, B)\right) \quad (4)$$

dove $\text{dist}(x_i, B) = [\text{dist}(x_i, b_1), \ldots, \text{dist}(x_i, b_M)]^T$, e $\text{dist}(x_i, b_j)$ è la distanza euclidea tra x_i e b_j.
Il vincolo $\mathbf{1}^T c_i = 1$ segue i requisiti di invarianza per spostamento del codice LLC. Si noti che il codice LLC nell'Eq. 3 non è sparso nel senso della norma 0, ma è sparso nel senso che la soluzione ha solo pochi valori significativi. In pratica, semplicemente impostiamo a zero quei coefficienti piccoli mediante soglia.
Il termine di regolarizzazione della località comporta alcune proprietà interessanti:

- **Miglior ricostruzione**: ogni descrittore è più accuratamente rappresentato in LLC da multiple basi, condividendo le quali, vengono catturate correlazioni fra descrittori simili.
- **Local smooth sparsity**: a causa dell'eccessiva completezza del codebook, il processo SC potrebbe selezionare basi piuttosto diverse per patch simili per favorire la sparsità, perdendo così le correlazioni tra i codici. LLC assicura che patch simili avranno coefficienti simili.
- **Soluzione analitica**: nella pratica, le performance nel calcolo di LLC possono essere molto buone e la codifica rapida.

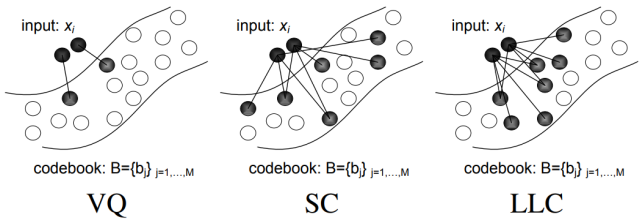


Figure 5: Confronto fra VQ, SC e LLC: si può notare come applicare LLC porta a considerare un numero sufficiente di caratteristiche, nello spazio identificato dai codebooks, vicine all'input.

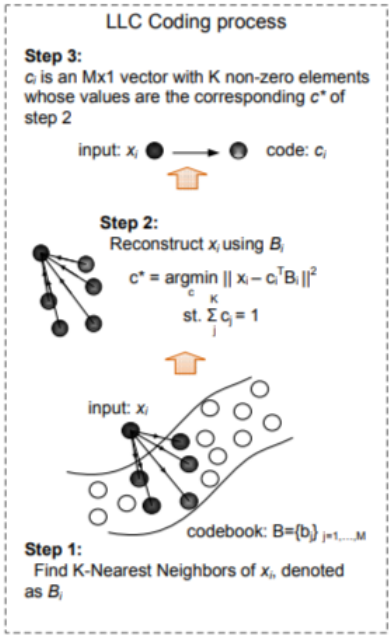


Figure 6: Processo di encoding LLC

6 Fase 4: Spatio-Temporal Pooling

La codifica è seguita da una fase che rappresenta un adattamento del concetto di 'Spatial Pyramid Matching' (SPM) nell'ambito dei video. Questa tecnica comprende l'uso dell'informazione sia temporale che spaziale per la codifica delle relazioni tra queste due dimensioni.
Si tratta di una piramide spazio-temporale che partiziona un video in una griglia 3D e effettua il pooling (in questo caso max pooling) di ognuna delle $2l \times 2l \times l$ con $l = 0, 1, 2$ sottoregioni. Similmente a SPM, un video è prima visto nella sua interezza, poi, al secondo livello è segmentato spazialmente in 4 frammenti senza alcuna segmentazione temporale. Al terzo livello ogni parte, così come era al precedente livello, è suddivisa in 4 sottoregioni spaziali e 2 sottoregioni temporali.
Il descrittore finale è costituito da un istogramma che raccoglie i contributi di ogni sottoregione.

7 Experimental setup

7.1 KTH dataset

Uno dei dataset presi in considerazione è KTH, uno dei set di dati più noti per il task di activity recognition. Contiene sei categorie di azioni: walking, jogging, running, boxing, hand waving e hand clapping. Ogni azione viene eseguita da 25 individui diversi e l'ambientazione viene sistematicamente modificata. Le variazioni includono: azione eseguita outdoor (s1), outdoor con variazione di scala (s2), outdoor con abiti diversi (s3) e indoor (s4). Questi cambiamenti mettono alla prova le capacità degli algoritmi di identificare attività indipendentemente dallo sfondo, dall'aspetto degli attori e dalla loro dimensione.

Figure 7: Alcuni esempi di dati estratti dal KTH dataset

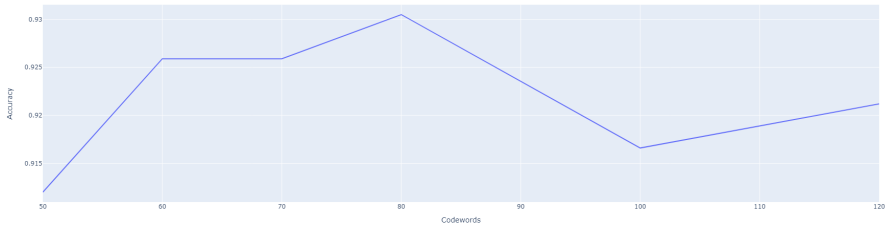
7.2 Hollywood2 dataset

Si tratta di un dataset piuttosto corposo: contiene campioni video tratti da 32 film. Ogni sample è etichettato in base a una o più delle 8 classi: AnswerPhone, GetOutCar, HandShake, HugPerson, Kiss, SitDown, SitUp, StandUp. Hollywood2 è diviso in un test set ottenuto da 20 film e due training set ottenuti da 12 film diversi. Il set di training automatico è ricavato utilizzando annotazioni automatica applicate mediante script e contiene 233 esempi di video con circa il 60% di etichette corrette. Il set di formazione clean contiene 219 esempi con etichette verificate manualmente.
Cito per completezza Hollywood2, anche se, durante lo svolgimento dell'elaborato, l'estrattore non è riuscito a ricavare abbastanza feature dai video da poter permettermi di condurre esperimenti su questo set, analogamente a quanto descritto nel paper preso in esame per il progetto; per questo, nonostante il tentativo di seguire pedissequamente lo studio originale, questa riproduzione si è concentrata interamente su KTH.

Figure 8: Alcuni esempi tratti da Hoollywood dataset

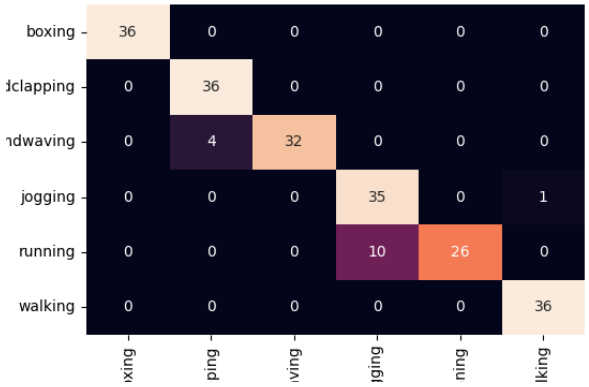
7.3 Risultati sperimentali

In questa sezione vengono analizzati i risultati degli esperimenti (che, come si è detto, riguardano KTH dataset).
Come test set sono selezionati 9 soggetti (2, 3, 5, 6, 7, 8, 9, 10 e 22) mentre i rimanenti 16 vengono impiegati per il training set.



La figura mostra sull'asse delle X il numero di codewords utilizzate per la rappresentazione Bag-of-features, mentre sull'asse delle Y si trova l'accuratezza. Le migliori performances si sono registrate con un numero totale di codewords di 480 (cioè 6 classi di azioni, 80-codes per classe), un risultato equivalente a quello ottenuto dallo studio originale, il quale, con questi dati, aveva già dimostrato di aver superato i metodi allo stato dell'arte e come fosse migliore un approccio a più codebook rispetto ad un unico macro-dizionario.
Viene sottolineata in particolare la superiorità sulla classica BoW (il 6% di miglioramento delle performances), che seleziona un set di 100.000 random features per ricavare un codebook unico, con 4000 codewords e K-means utilizzato come algoritmo di clustering, VQ come tecnica per la quantizzazione e infine SVM come classificatore.

Per ognuno dei test viene presentata anche la matrice di confusione, rappresentazione dell'accuratezza di classificazione statistica: ogni colonna rappresenta i valori predetti, mentre ogni riga i valori reali.
Questa la rappresentazione per la configurazione che ha ottenuto le prestazioni migliori: Si può notare come gli errori si concentrino sulle azioni simili fra



loro: ad esempio, 4 video che apparterrebbero alla classe "handwaving" vengono etichettati come "handclapping" e per 10 della classe "running" viene predetta "jogging", errori "ammissibili" per dati con cui ci si può aspettare che un qualunque approccio possa faticare, visto che si parla di attività che sono strettamente legate fra loro.
Questo ci permette di constatare ulteriormente la bontà della soluzione progettata per risolvere il problema, che anche con questa verifica dimostra di essere un approccio di successo.

8 Conclusioni

Questo elaborato mi ha dato l'occasione di sperimentare concretamente un approccio di cui non avevo avuto modo di verificare le potenzialità. Mi ritengo soddisfatto per essere riuscito a ripetere l'esperienza del team che aveva realizzato il lavoro originale e ad ottenere gli stessi risultati.
Quanto analizzato rimane effettivamente progredibile, mediante l'impiego di nuovi descrittori e nuovi modelli: le prestazioni infatti potrebbero effettivamente beneficiare dall'applicazione di diverse tecniche di classificazione e da modellazioni delle relazioni spazio-temporali alternative, che potrebbero estendere l'utilizzo della tecnica a diversi contesti che esulano dal semplice riconoscimento dell'attività.

References

Kläser, Marszalek, and Schmid. “A spatio-temporal descriptor based on 3d-gradients,” 2008.

Laptev. “On space-time interest points,” 2005.

Laptev, Marszałek, Schmid, and Rozenfeld. “Learning realistic human actions from movies,” 2008.

Lazebnik, Schmid, and Ponce. “Beyond bags of features: spatial pyramid matching for recognizing natural scene categories,” 2006.

Marszalek, Laptev, and Schmid. “Actions in Context,” 2009.

Schuldt, Laptev, and Caputo. “Recognizing human actions: a local svm approach,” 2004.

Umakanthan, Denman, Fookes, and Sridharan. “Multiple Instance Dictionary Learning for Activity Representation,” 2014.

Wang, Wang, Bai, Liu, and Tu. “Max-margin multiple instance dictionary learning,” 2013.

Wang, Yang, Yu, Lv, Huang, and Gong. “Locality constrained linear coding for image classification,” 2010.

Yang, Gong, and Huang. “Linear spatial pyramid matching using sparse coding for image classification,” 2009.

Yu, Zhang, and Gong. “Advances in Neural Information Processing Systems,” 2009.