UNIVERSITÀ DEGLI STUDI DI FIRENZE
Facoltà di Ingegneria - Dipartimento di Sistemi e Informatica

Tesi di laurea in Ingegneria Informatica Specialistica

CLASSIFICAZIONE DI IMMAGINI CON METODI MULTIMODALI GENERATIVO-DISCRIMINATIVI

IMAGE CLASSIFICATION USING MULTIMODAL GENERATIVE-DISCRIMINATIVE METHODS

Candidati
Andrea Matteo Serain

Benito Fabio Zaccone

Relatori
Prof. Alberto Del Bimbo

Ing. Marco Bertini

Correlatori
Ing. Lamberto Ballan

Ing. Giuseppe Serra

Anno Accademico 2010-2011

Indice

Introduzione

Lo studio e la classificazione delle immagini si avvalgono ormai da anni del contributo fornito dai diversi approcci di analisi e dalle numerose tecniche di descrizione dei dati multimediali. La letteratura fornisce una vasta gamma di soluzioni al problema, con metodologie che si distinguono per il modo in cui l'informazione viene ricercata all'interno del documento e per lo schema attraverso cui essa viene rappresentata nei software di elaborazione.
Si è visto negli anni come una base di partenza possa essere la ricerca dell'informazione a livello locale, attraverso la selezione di un insieme di elementi altamente descrittivi come i *punti salienti* in grado di catturare una parte importante del contenuto informativo: questo principio, insieme ad alcune tecniche mutuate da altre discipline come ad esempio l'*Information Retrieval*, ha condotto alla definizione di una vasta gamma di approcci diversi per la classificazione delle immagini.
La ricerca continua di miglioramenti volti a superare alcuni limiti dei metodi di base ha portato ad affrontare il problema della descrizione delle immagini da diversi punti di vista: da un lato la mancanza di localizzazione spaziale dei punti salienti ha suggerito l'utilizzo delle informazioni di posizione per la costruzione di modelli che fornissero un ordine su cui vincolare la descrizione delle immagini; dall'altro la necessità di una semantica diretta degli elementi informativi analizzati dai metodi classici ha indirizzato la ricerca verso un uso di modelli matematici più complessi, che consentissero di elevare la descrizione dei dati ad un livello semantico superiore.
Parallelamente agli approcci nati dalla ricerca di feature locali, metodologie alternative si sono sviluppate su tecniche di descrizione dei documenti basate su un'analisi di tipo globale. Il processo è stato motivato dalla convinzione

che l'interpretazione percettiva di una scena visiva possa prescindere dalla conoscenza dei dettagli di basso livello, ma, al contrario, possa essere avvantaggiata dalla decisione di catturare esclusivamente gli aspetti essenziali della rappresentazione.

Con queste premesse è risultato interessante in questo lavoro osservare il comportamento di diversi approcci proposti dalla letteratura, sia in test che prevedono l'utilizzo singolo delle diverse tecniche in configurazioni *unimodali*, sia in configurazioni *multimodali* che prevedono l'impiego congiunto di due o più metodologie, ponendo particolarmente attenzione all'eventuale vantaggio di avvalersi di soluzioni semanticamente più raffinate, oppure di soluzioni ibride che traggono giovamento dalla presunta complementarità di tecniche diverse fuse in pipeline multimodali.
Questi argomenti di interesse sono stati esaminati attraverso una serie di esperimenti condotti prima di tutto su una collezione di immagini di riferimento per la letteratura scientifica, con lo scopo di validare le prestazioni rispetto allo stato dell'arte.
In una seconda fase del lavoro è risultato interessante assecondare la tendenza di questi ultimi anni di spostare l'attenzione sulla classificazione di documenti realistici e condivisi pubblicamente sui più popolari social network. Cercando contenuti liberamente disponibili su Flickr, è stato creato un nuovo dataset sul modello di quello di riferimento. L'analisi di immagini realistiche piuttosto che di documenti spesso creati appositamente per scopi scientifici ha da un lato aumentato il livello di difficoltà del problema della classificazione visuale, mentre dall'altro ha permesso di integrare la descrizione dei contenuti visuali con l'analisi dei metadati testuali inseriti dagli utenti del network. Ciò ha permesso di estendere le configurazioni multimodali con informazioni di natura diversa ma evidentemente in reciproca relazione semantica. Nella fase finale degli esperimenti il sistema di classificazione basato sull'analisi delle informazioni visuali è stato quindi ampliato con l'analisi del contributo testuale relativo a ciascuna immagine, in configurazioni multimodali che combinano insieme approcci differenti ed informazioni complementari.

Capitolo 1

Stato dell'arte

I modelli *feature based* sono ad oggi tra i più popolari per la descrizione di immagini. L'obiettivo di questa tipologia di tecniche è individuare parti dell'immagine che possano essere considerate rilevanti secondo qualche criterio: in genere si desidera selezionare porzioni della scena che siano rappresentative del contenuto informativo del documento, o che possano darne nell'insieme un'adeguata descrizione in termini visuali.
Ciascuna delle feature locali individuate viene descritta generalmente tramite un vettore di valori numerici, in modo da consentire un suo impiego in operazioni di classificazione o Information Retrieval che coinvolgano la stima del grado di similarità tra feature diverse. Attraverso tecniche di clustering è poi possibile raggruppare tra loro feature simili, in modo da ottenere un modello di descrizione di più alto livello. Questo tipo di approccio, mutuato dall'analisi del testo, è il cosiddetto *bag of words*: un "documento", sia esso di natura testuale o un'immagine, viene descritto solo in base alla frequenza di un certo numero di "parole" al suo interno, senza tener conto della loro posizione in esso. Nel caso testuale le "parole" sono ben definite dal linguaggio usato; nel caso visuale esse sono in genere associate ai rappresentati di classi di equivalenza di feature locali.

Esistono molte soluzioni per la classificazione di immagini basate sul modello bag of words. Per incrementare le prestazioni il semplice modello base

viene però sempre più spesso affiancato da altre tecniche di rappresentazione, che consentano di creare modelli semanticamente più rilevanti.
Ad esempio di recente sono state elaborate svariate implementazioni che impiegano l'uso di "modelli a topic latenti" come pLSA o LDA [5] costruiti sulla base della distribuzione delle parole visuali. Grazie ad essi è possibile ottenere una riduzione della dimensionalità nella rappresentazione delle immagini ed al tempo stesso inferire informazioni di più alto livello. Questo genere di metodi viene impiegato con successo in applicazioni di Information Retrieval, classificazione e segmentazione.
Da non sottovalutare inoltre il possibile apporto che può derivare dalla componente *social* dei portali che raccolgono dati multimediali creati da utenti comuni: sfruttando queste risorse è possibile costruire dataset realistici ed utilizzare meta-informazioni per agevolare i processi di classificazione; l'uso di descrizioni di natura differente, inoltre, consente di implementare algoritmi di classificazione multimodali.

1.1 Modelli a topic latenti

Gli algoritmi basati sulla ricerca di *topic latenti* a partire da una descrizione di tipo bag of words visuali di immagini sono stati utilizzati sia per applicazioni di classificazione che di segmentazione ed image retrieval.

Nel lavoro di Fei-Fei e Perona [11] viene presentato un metodo di apprendimento e riconoscimento di categorie in scene naturali in cui viene usato LDA per ricavare un livello intermedio di rappresentazione: le parole visuali vengono raggruppate in *classi* (o temi) ed i temi sono poi usati per assegnare una *categoria* all'immagine. Utilizzando un dataset di fotografie di elementi naturali suddivise in 13 categorie, gli autori hanno confrontato le prestazioni ottenute variando sia la tipologia di rilevatori (campionamento denso a passo costante, campionamento casuale, regioni di rilevanza di Kadir & Brady, Differences of Gaussians) che quella dei descrittori (SIFT, livelli di grigio su una finestra 11x11). Per la creazione del vocabolario visuale è stato impiegato l'algoritmo K-means. I risultati mostrano che la soluzione migliore è

quella che utilizza punti derivanti da un campionamento denso descritti da un SIFT-128, che arriva ad un'accuratezza di classificazione del 65,2%.

Il lavoro presentato da Horster *et al.* [10] introduce un metodo di image retrieval basato sul contenuto per mezzo di modelli LDA, utilizzando per i test un dataset di 246.000 immagini divise in 12 categorie; tale insieme è stato costruito a partire da ricerche di un numero limitato di tag tra i contenuti multimediali pubblicamente disponibili su Flickr. Vengono comparate le prestazioni ottenute utilizzando misure di distanza diverse per la valutazione della similarità: coseno, L1, divergenza Jensen-Shannon, likelihood. Il metodo LDA viene inoltre confrontato con una rappresentazione basata su pLSA.
I risultati ottenuti mostrano che per valutare la similarità la misura che offre le migliori prestazioni è quella basata sulla likelihood, ovvero indicizzare i documenti in base alla probabilità che il modello derivante dalla distribuzione di parole visuali in ciascuno di essi possa generare la query; in caso di dataset di grandi dimensioni sembra preferibile però il secondo miglior metodo in termini di prestazioni, ovvero quello basato sulla divergenza JS. I test comparativi tra i due modelli a topic latenti portano a concludere che il punteggio ottenuto dai modelli pLSA è sempre significativamente inferiore rispetto alle prestazioni ottenute da LDA.

1.1.1 Classificazione di scene tramite pLSA

Nel lavoro di Anna Bosh *et al.* del 2006 [6] viene proposta una soluzione per la classificazione di immagini che usa un metodo completamente non supervisionato basato su pLSA. L'idea di base è quella di trasporre le tecniche di ricerca di topic latenti da un ambito testuale ad uno visuale: le immagini vengono associate al concetto di "documento", mentre i "topic" sono rappresentati dagli oggetti raffigurati in esse; le "parole visuali" sono ottenute tramite un processo di quantizzazione di descrittori SIFT-like. Ogni immagine viene descritta dal vettore di distribuzione dei topic al suo interno.
La creazione del dizionario visuale avviene tramite l'estrazione di features

locali SIFT-like nelle immagini, che vengono poi quantizzate attraverso K-means per ottenere le parole visuali. Il modello pLSA viene calcolato sulle immagini di addestramento; in base ad esso vegono determiate le distribuzioni dei topic anche per immagini incognite in modo da minimizzare la divergenza Kullback-Leibler[1] rispetto ai dati di training. La categoria attribuita ad ogni nuova immagine è quella più presente tra le k-Nearest-Neighbours del training set etichettate.
Per la creazione delle parole visuali sono state anche in questo caso testate soluzioni diverse per la ricerca di feature locali: tra le varie metodologie testate quella che ha ottenuto le migliori prestazioni è un SIFT denso calcolato sui tre canali HSV su quattro scale. Gli autori hanno testato il metodo proposto sul dataset di scene naturali proposto in [11], ottenendo risultati migliori di quelli presentati nell'articolo originale.

In una versione successiva dello stesso articolo [7] gli autori estendono il metodo proposto distinguendo due fasi di apprendimento successive: durante la prima fase, di tipo *generativo*, viene costruito un modello a topic latenti a partire dalle occorrenze delle visual words nelle immagini; sulla base della descrizione a livello immagine così ottenuta e sfruttando la conoscenza a priori delle etichette delle immagini di training viene poi messa in atto una fase di learning di tipo *discriminativo* addestrando una macchina SVM. Questa soluzione si differenzia dalla precedente in quanto l'uso di una macchina SVM modifica a parte finale del processo di classificazione, sostituendosi di fatto al k-NN. La struttura aggiornata è mostrata in figura 1.1.
Gli autori hanno eseguito test comparativi sul dataset [11] per confrontare le prestazioni di pLSA e BoW classico nelle due varianti, quella generativa "pura" basata su k-NN e quella ibrida generativa/discriminativa che fa uso di SVM. Per il classificatore SVM viene usato un kernel χ^2 esponenziale per il modello BoW, mentre per pLSA uno esponenziale basato sulla distanza euclidea. Per incrementare ulteriormente le prestazioni sono state introdotte delle modifiche al metodo basato su pLSA, in modo da prendere in considerazione informazioni sulla posizione. Sono stati messi a confronto metodi

[1] misura della differenza tra due distribuzioni di probabilità

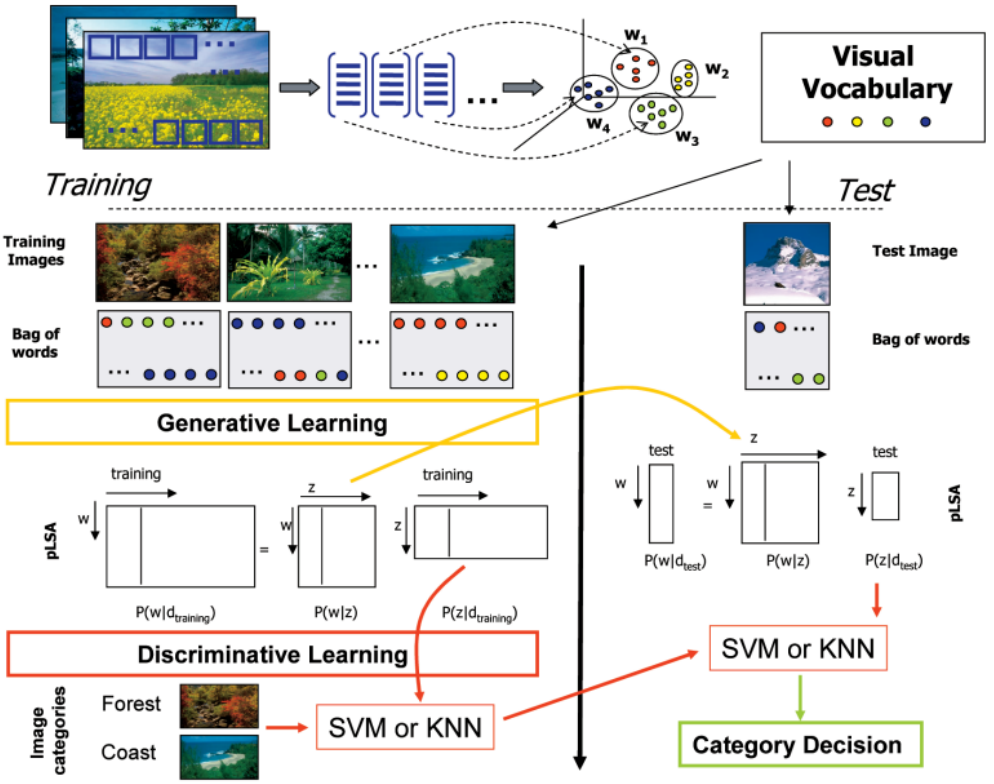


Figura 1.1. Schema della versione ibrida del metodo di classificazione tramite pLSA

diversi: alcuni che prevedono la semplice aggiunta delle informazioni di posizione ai feature vector dei keypoint, denominati *xy-pLSA* e *ABS-pLSA*; altri che invece provano ad integrare rappresentazioni strutturate a piramide spaziale con l'informazione ottenibile tramite topic latenti. Un'implementazione dello Spatial Pyramid Matching (SPM, [37]) è stata confrontata con l'innovativo *Spatial Pyramid - pLSA* (*SP-pLSA*). Test eseguiti su Caltech-101 hanno mostrato che quest'ultimo metodo offre prestazioni migliori di tutti quelli precedentemente descritti.

1.1.2 Segmentazione

I modelli a topic latenti sono stati impiegati anche in molti lavori che mirano alla classificazione delle immagini tramite la loro segmentazione, in modo da individuare oggetti semanticamente rilevanti al loro interno.

Ad esempio Sivic *et al.* in [36] presentano un metodo non supervisionato di classificazione di scene mediante pLSA che estende i metodi già descritti introducendo il concetto di *doublets*: coppie di parole che occorrono nelle stesse immagini in regioni spaziali contigue, escluse regioni con elevata sovrapposizione. In questo modo viene introdotto il concetto di specificità spaziale nella descrizione degli oggetti, in modo da migliorare i risultati della segmentazione. Le parole visuali sono ottenute a partire da descrittori SIFT e regioni MSER. Un'analisi della distribuzione dei topic nelle diverse immagini porta gli autori ad osservare che le parole più rappresentative per ciascuna categoria descrivono regioni dell'immagine rilevanti anche dal punto di vista semantico per identificare gli elementi che la contraddistinguono. Ad esempio nella categoria "volti" sono particolarmente rilevanti le parole visuali associate agli occhi.
La classificazione avviene come in precedenza descrivendo ogni immagine in base alla percentuale di topic presenti in essa; la successiva segmentazione avviene tramite una procedura di allineamento tra le parole visuali e le regioni da individuare nell'immagine. Test eseguiti su Caltech-101 hanno mostrato che l'uso di *doublets* porta un miglioramento del 20% delle prestazioni di classificazione.

Un framework che combina LDA per il riconoscimento e un modello ibrido parametrico-non parametrico per la segmentazione è la soluzione adottata da Andreetto *et al.* in [24]. L'obiettivo del progetto è la scoperta di relazioni tra *segmenti* in una collezione di immagini, intesi come parti di un oggetto. Si propone infatti di apprendere dei *categorical segments* svolgendo simultaneamente segmentazione, scoperta di corrispondenze tra segmenti appartenenti ad immagini diverse e riconoscimento. In questo caso le visual word sono

ottenute a partire da un campionamento denso in cui a ciascun punto è associato un vettore di feature che comprende la posizione e i valori RGB, mentre LDA viene usato per modellare la distribuzione dei topic all'interno di ogni segmento.

Arrivare alla comprensione totale della scena è l'obiettivo del lavoro presentato da Li e Fei-Fei in [17]: in esso gli autori propongono un modello generativo gerarchico basato su topic latenti che consente di classificare una scena, riconoscere e segmentare ciascun suo componente visuale ed annotare l'immagine con una lista di tag. In questo senso si vuole arrivare ad una descrizione dell'immagine con un modello congiunto che derivi sia da un modello di tipo visuale che da uno di tipo testuale.
Il modello proposto può riconoscere e segmentare oggetti multipli all'interno delle immagini; si pone inoltre come il primo tentativo di ottenere una descrizione globale per l'apprendimento automatico a partire da immagini e tag ottenute dalla rete, ad esempio da Flickr. Si introduce inoltre una soluzione per identificare possibili tag erroneamente assegnati. I test sono stati eseguiti su un dataset di otto classi rappresentati scene sportive: 800 elementi per classe, di cui 200 usati per i test. I tag associati ad esse sono stati filtrati tramite un opportuno algoritmo: dopo aver eliminato quelli che non si riferiscono ad "entità fisiche" secondo WordNet, i rimanenti vengono raggruppati in base ai *synset* specificati sullo stesso portale. Dei 1256 tag unici ottenuti al termine di questa fase di preprocessing, sono stati utilizzati per l'algoritmo solo i 30 più frequenti.

1.2 Social knowledge

La quasi totalità delle soluzioni proposte finora utilizza dataset costruiti artificialmente per mettere alla prova le metodologie proposte, con immagini scelte appositamente per la creazione dei dataset dagli stessi ricercatori che si occupano del progetto, etichettate spesso da utenti specializzati. Vista però la grande diffusione di contenuti multimediali su Internet e soprattuto il continuo accrescimento di portali per la condivisione di immagini come Flickr

(`www.flickr.com`) viene spontaneo chiedersi in quale misura questa quantità di informazioni direttamente disponibile ed in continua crescita possa essere impiegata a scopi di ricerca.

Come già visto in [17], l'uso di questo genere di immagini consente anche di impiegare le informazioni derivanti dall'analisi dei "tag", ovvero le brevi descrizioni associate alle stesse dai loro autori. In prima istanza si potrebbe essere portati ad associare il *tagging* sociale alla classificazione accurata compiuta dai ricercatori: in questo modo l'uso di questo genere di immagini consentirebbe di creare dataset in cui l'operazione di etichettatura diverrebbe superflua. I tag costituirebbero inoltre una possibile sorgente informativa aggiuntiva direttamente associabile all'immagine. Ma quali benefici porterebbe questa soluzione? Quanto sono realmente affidabili i tag in applicazioni di classificazione? Nel lavoro di Setz e Snoek del 2009 [35] viene mostrato come i tag siano spesso ambigui e altamente personalizzati, oltre a poter contenere errori in alcuni casi. I ricercatori mostrano che però dopo una prima fase di filtraggio e disambiguazione l'uso di questa informazione aggiuntiva porta ad un miglioramento delle prestazioni di classificazione, soprattutto per concetti che sono visivamente consistenti al variare del dominio di ricerca delle immagini del dataset. In [18] Li e Snoek introducono l'idea di una sostituzione del social tagging alle procedure di *expert labeling* per creare esempi negativi per la classificazione binaria. Vengono confrontati diversi scenari di apprendimento a due classi (esempi positivi - esempi negativi): nel primo scenario entrambi gli insiemi sono costituiti con immagini etichettate da esperti; nel secondo l'insieme degli esempi negativi è ottenuto tramite query su Flickr basate sui tag. Identificato l'insieme dei sinonimi del concetto che si intende classificare, l'insieme dei negativi viene costruito rimuovendo da una collezione di immagini scaricate da Flickr quelle che sono associate ad almeno un sinonimo. La riduzione di prestazioni usando i *free negative examples* si attesta intorno al 4%.

Il problema principale in caso di utilizzo di una sorgente social per la creazione del dataset diventa dunque trovare un modo per elaborare l'insie-

me dei tag in modo da renderlo non ambiguo. In [35] il processo è eseguito volutamente manualmente, ma altri autori suggeriscono molteplici soluzioni di tipo automatico o semi-automatico.
Una soluzione interessante può essere quella introdotta dal cosiddetto *ESP game* [2]: basandosi sull'assunto che un tag affidabile sarà scelto da più utenti, il gioco consiste nel mostrare a due utenti, scelti casualmente, la stessa immagine; gli utenti, che non possono comunicare tra di loro, devono scrivere un elenco di termini che secondo loro descrivano efficacemente quanto rappresentato; non appena lo stesso termine viene usato da entrambi i giocatori, ad ognuno di essi viene assegnato un punto ed il gioco prosegue con l'immagine successiva. Se un termine viene assegnato ad un'immagine un numero sufficiente di volte, viene aggiunto ai tag. I punti di forza di questa idea sono legati all'affidabilità di annotazioni compiute da utenti diversi,supponendo che questi descrivano in modo oggettivo il contenuto visuale dell'immagine.
L'algoritmo proposto da Kennedy *et al.* [21], mira ad applicare i concetti introdotti nell'ESP game ad un dataset di immagini ottenuto da Flickr. In questo caso viene fatto notare come i fotografi, che hanno realizzato le immagini condivise online, possano essere considerati quasi al pari di annotatori esperti rispetto ai loro lavori; quindi immagini visivamente simili saranno molto probabilmente annotate dai loro autori con tag analoghi. Sembra dunque sensato supporre che, analogamente al caso del gioco ESP, se due autori diversi usano gli stessi tag per descrivere immagini vicine nello spazio dei descrittori visuali, tali tag siano connessi col contenuto visivo delle immagini. Dato un insieme di immagini, ne vengono cercate altre che siano ad esse "simili" in base a feature visuali di basso livello, ma create da autori diversi e con alcuni tag in comune. Ai tag delle immagini originali viene quindi assegnato un punteggio, che sarà tanto più alto quanto più essi saranno frequenti nell'insieme creato.

Partendo da presupposti simili, Li e Snoek [19] introducono un metodo per assegnare un punteggio di rilevanza ai tag in base alla loro occorrenza all'interno dell'insieme dei vicini di un'immagine creato per mezzo delle sue feature visuali. In particolare viene introdotta una misura di *tag relevance*

che tiene conto della distribuzione dei tag sia nell'insieme dei vicini che a livello globale nell'intero dataset; la formula finale è pari alla differenza tra un punteggio derivante dal conteggio delle occorrenze del tag nell'insieme dei vicini e la sua frequenza a priori.
Sono stati condotti esperimenti di image retrieval basata sui tag, utilizzando una funzione punteggio in cui alla frequenza dei termini viene sostituito il punteggio di tag relevance introdotto. In modo analogo gli autori presentano anche approcci di tag suggestion sia per immagini etichettate che non etichettate.

La soluzione di calcolo della rilevanza dei tag presentata in [19] è stata ad esempio utilizzata da Ballan *et al.* in [16] per l'assegnazione automatica di tag a parti di sequenze video. Prendendo in considerazione un video caricato su uno dei più popolari siti di condivisione di dati multimediali (YouTube, Facebook, Vimeo), vengono considerati separatamente i sui tag ed alcuni sui keyframe.
I tag sono usati come parole chiave per ricerche su Flickr, in modo da ottenere un insieme di immagini che siano etichettate con gli stessi termini; uno step di clustering consente di raggruppare immagini visivamente simili. Per ogni keyframre estratto dal video, si cercano immagini ad esso vicine nei cluster costruiti come descritto sopra, basandosi sempre su opportune misure di distanza nello spazio delle caratteristiche visuali. Sulla base dell'insieme di vicini così ottenuto si crea una lista di tag suggeriti per il frame facendo uso della funzione di tag relevance.

1.2.1 Metodi multimodali

La combinazione delle descrizioni derivanti da diverse unimodal feature, specialmente in caso identifichino caratteristiche dissimili del contenuto informativo, possono portare ad un consistente incremento delle prestazioni rispetto ai metodi di partenza. Le tecniche di fusione possono a grandi linee essere divise in due categorie: *early fusion*, in caso la combinazione avvenga prima del processo di apprendimento; *late fusion* in caso contrario. Il secondo me-

todo fornisce prestazioni migliori nella maggior parte dei casi [9], per questo nel presente lavoro sono stati considerati approcci di questo tipo.
Un'ulteriore classificazione dei diversi approcci deriva dalla tipologia di feature ad essere combinate tra loro: in alcuni casi ad esempio vengono accostate solo caratteristiche che derivano da un'analisi visiva della scena rappresentata, facendo attenzione ad utilizzare descrittori che vadano a descrivere aspetti eterogenei; in caso sia possibile associare all'immagine anche informazioni di altra natura, ad esempio di tipo testuale, può essere possibile includere anche i modelli generati a partire da esse nell'analisi mutlimodale.

Un esempio del primo tipo, che fa quindi uso solo di feature visuali, è presentato in [32] da Gehler e Nowozin: in tale lavoro gli autori confrontano metodologie diverse di apprendimento multimodale basato sui metodi kernel. In particolare i metodi cosiddetti "baseline", ovvero media aritmetica (*average*) e media geometrica (*product*) dei kernel derivanti dalle diverse feature, vengono comparati con un MKL (Multiple Kernel Learning) puro e con alcune tecniche di *LPBoost* (linear programming boosting).
Le feature visuali considerate sono: descrittori SIFT a griglia densa su quattro scale, un istogramma delle direzioni del gradiente calcolate come output di un edge detector Canny, valori di covarianza locale di feature a livello pixel, *LPB* (pattern binari locali), funzioni di Gabor. Tutte le feature, tranne l'ultima, sono calcolati a più valori di scala, per un totale di 39 kernel utilizzati nei diversi processi di fusione.
I risultati ottenuti da test eseguiti anche su Caltech-101 mostrano che le semplici soluzioni baseline, molto più veloci delle altre tecniche di apprendimento, offrono ottime prestazioni; solo le tecniche basate sul boosting raggiungono performance di miglior livello su tutti i test.

Un approccio diverso è quello presentato da Lienhart *et al.* in [33]. Gli autori propongono un modello a topic latenti basato su pLSA per realizzare un'applicazione di Information Retrieval che fa uso sia di informazioni derivanti da analisi visuale delle immagini che dei dati ottenuti dai tag ad esse associati. Partendo da considerazioni relative ai processi di apprendimento

umani, viene introdotto un metodo *multimodale multilivello*: vengono appresi separatamente un modello a topic latenti a partire dalle feature visuali ed uno derivato dai tag; le due descrizioni così ottenute vengono fuse in un unico vettore, usato come input per un nuovo passo di apprendimento pLSA che fornisce una descrizione di più alto livello dell'immagine.
Il modello visuale viene costruito in modo simile a quanto fatto in lavori analoghi, come ad esempio in [6]: le feature puntuali individuate nelle immagini vengono quantizzate per formare un codebook di visual word; costruita una tabella di co-occorrenza delle parole in ogni documento, l'applicazione di pLSA consente di ottenere un descrittore per ogni immagine, costituito dalla frequenza dei diversi topic al suo interno. Per utilizzare i tag, essi vengono anzitutto filtrati per eliminare le stopwords; in seguito una ricerca su WordNet consente di consolidare ulteriormente l'insieme delle parole scelte: questa analisi inoltre consente di includere sinonimi ed iperonimi fino al terzo livello al dizionario finale. Sulla base dei token così ottenuti viene creato un modello a topic latenti in modo analogo al caso visuale.
I test compiuti sul dataset introdotto in [10] mostrano un incremento di prestazioni del 19% rispetto al singolo metodo unimodale basato sulle feature visuali.

Un altro approccio del secondo tipo è stato introdotto in [28] da Rasiwasia *et al.*: in questo caso ci si pone il problema di determinare il grado di similarità tra un documento testuale ed un'immagine, e viceversa. Per far ciò viene proposto un metodo che fa uso di metodi diversi per ottenere rappresentazioni ad elevati livelli di astrazione a partire da elementi eterogenei. Tecniche di *correlation matching*, come latent semantic indexing (LSI) o principal component analysis (PCA), vengono integrate con strategie di *semantic matching* come LDA.

1.2.2 Insiemi di immagini *real-world*

Come si è visto, possono derivare indubbi vantaggi dall'uso di un dataset proveniente da una fonte *social*. Primo fra tutti l'immediata disponibilità

di meta-informazioni che possono essere associate alle immagini, da utilizzare durante le fasi di creazione dei modelli matematici per accrescere la loro accuratezza. Realizzare soluzioni che ben si adattano a questi insiemi *real-world*, inoltre, offre la possibilità di usufruire dell'immensa quantità di dati liberamente disponibile online, ad oggi in continua crescita.
A esempio Hörster *et al.* in [10] hanno testato il metodo presentato su un dataset di circa 246k immagini, scelte tra quelle postate su Flickr prima del Settembre 2006 ed etichettate come *geotagged* insieme ad almeno uno dei seguenti tag: *sanfrancisco*, *beach* e *tokyo*. Tra tutte quelle ottenute sono state conservate solo quelle che contevano almeno uno dei seguenti tag: *wildlife*, *animal*, *animals*, *cat*, *cats*, *dog*, *dogs*, *bird*, *birds*, *flower*, *flowers*, *graffiti*, *sign*, *signs*, *surf*, *surfing*, *night*, *food*, *building*, *buildings*, *goldengate*, *goldengate-bridge*, o *baseball*. In totale sono state dunque costruite 12 categorie, con una media di 20.500 immagini per classe.

Un dataset più recente, realizzato utilizzando le immagini rilasciate sotto licenza Creative Commons su Flickr e costituito da un numero inferiore di immagini, ma con maggiore variabilità in termini di tag, denominato **MIR Flickr** [27].
Secondo gli autori il dataset ideale dovrebbe soddisfare i seguenti requisiti:

- essere **rappresentativo** per l'area di interesse: in caso di dataset derivati da fonti *social*, dovrebbe essere realizzato a partire dal contributo di migliaia di utenti diversi;
- disponibilità di risultati di base (**ground truth**), da poter usare come metro di paragone per test futuri;
- essere facilmente **accessibile** e liberamente **condivisibile**;
- disponibilità di risultati per **test standardizzati**.

Nella prima versione esso comprende 25000 immagini ottenute tramite l'API di Flickr, cercando foto scattate tra Marzo 2007 ed Giugno 2008. Alla creazione della collezione contribuiscono 9862 autori diversi, dei quali 5566

sono rappresentati nell'insieme da una sola immagine. Il numero medio di tag è 8.94 per immagine, con 1386 tag che compaiono in almeno 20 immagini. I tag più frequenti sono: *sky* (845 img.), *water* (641), *portrait* (623), *night* (621), *nature* (596), *sunset* (585), *clouds* (558), *flower* (510), *beach* (407), *landascape* (385). Oltre ai tag, per ogni immagini sono forniti i dati EXIF[2] dei parametri della fotocamera. Un gruppo di annotatori esperti ha raggruppato i tag in 11 argomenti di alto livello: *sky*, *water*, *people*, *night*, *plant life*, *animals*, *man-built structures*, *sunset*, *indoor*, *trasport*, *food*; alcuni di essi sono caratterizzati da sotto-argomenti associati ad etichette specifiche, per un totale di 27 classi. Su questo dataset sono state proposte tre tipologie di test standard: riconoscimento di "concetti" visuali; tag propagation; tag suggestion.
In un articolo successivo [26] sono stati forniti i risultati delle prestazioni di test di classificazione basati sugli argomenti assegnati ad ogni immagine, utilizzando sia descrizioni visuali globali di basso livello che l'analisi testuale dei tag. Viene inoltre introdotto il progetto di estendere il dataset ad 1 milione di immagini: questo lavoro si è concluso nel 2011[3], ed il risultato è stato usato nell'ImageCLEF 2011 workshop[4].

Anche Chua *et al.* hanno presentato in [8] un dataset di immagini costituito da elementi ottenuti da Flickr, denominato **NUS-WIDE**. Il numero complessivo di immagini è 269648, ognuna con i relativi metadati, per un totale di 5018 tag unici.
Per la descrizione sono stati utilizzati sei tipi diversi di feature di basso livello: istogramma di colore a 64-D, correlogramma di colore a 144-D, istogramma delle direzioni dei bordi 73-D, *wavelet textures* a 128-D, momenti di colore a blocchi 225-D, un modello bag of words a 500-D basato su descrittori SIFT. Il dataset è stato suddiviso secondo 81 concetti diversi.
In questo lavoro vengono introdotti alcuni metodi di annotazione e retrieval di immagini basati sul dataset presentato; un set di risultati baseline sono

[2] `http://exif.org`
[3] `http://press.liacs.nl/mirflickr/`
[4] `http://imageclef.org/2011`

stati infine ottenuti con una procedura di annotazione basata sul metodo k-NN.

Capitolo 2

Bag of Words

2.1 Classificazione di documenti testuali

Uno dei metodi più usati per la rappresentazione in ambito informatico di documenti testuali, sia per applicazioni di Information Retrieval che di classificazione, è il cosiddetto "bag of words". Questo modello prevede di rappresentare ogni documento attraverso la collezione dei termini in esso presenti, senza considerare le loro rispettive posizioni all'interno del testo. In entrambe le tipologie di problema la questione da affrontare è stabilire il grado di "similarità" tra un documento a cui non è ancora stata assegnata un'etichetta (*query*) e quelli che compongono il corpus già analizzato. A questo scopo è necessario stabilire dei metodi di:

- creazione del dizionario dei termini;
- rappresentazione dei documenti;
- creazione di un "modello" matematico di classificazione, basato sulle categorie dei documenti noti;
- classificazione di nuovi documenti, basata sul calcolo della "similarità" tra la query ed i documenti del modello.

2.1.1 Creazione del dizionario

Il dizionario dei termini non può derivare direttamente dall'elenco di tutte le parole che compaiono all'interno dei documenti: è necessario applicare delle operazioni di filtraggio per determinare gli elementi che hanno maggiore rilevanza dal punto di vista semantico.
Il processo di creazione dei *token* a partire da un documento testuale, ad esempio, deve anzitutto tener conto delle regole di punteggiatura e della rappresentazione di numeri, date, importi, oltre a stabilire come indicizzare termini provenienti da lingue diverse. Allo stesso modo è necessario stabilire in che modo trattare i sinonimi e se sia opportuno o meno convertire tutte le lettere in maiuscole.
La cosiddetta *regola di Zipf*, pubblicata dal linguista di Harvard George Kingsley Zipf, stabilisce che:

> *In un corpus di enunciati in linguaggio naturale, la frequenza di ogni parola è grosso modo inversamente proporzionale al suo grado (rank) nella tabella di frequenza. Quindi, la parola più frequente appare circa due volte più spesso della seconda parola più frequente, che appare due volte più spesso la quarta parola più frequente, ecc*

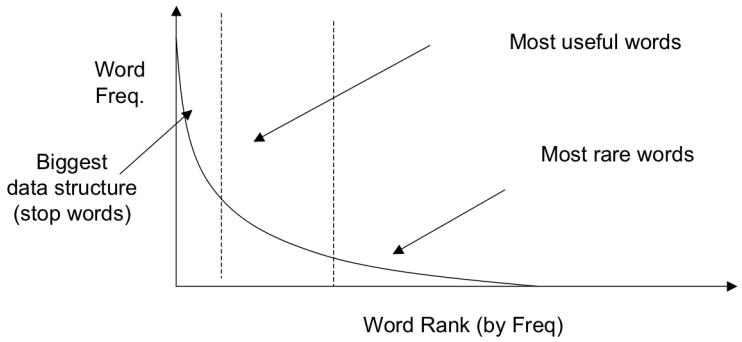


Figura 2.1. Relazione tra frequenza dei termini in un corpus e loro rilevanza semantica

In figura 2.1 viene rappresentato il concetto in forma grafica. Da notare che i termini che risultano di maggior interesse per le operazioni di classificazione

e ricerca sono quelli per i quali il valore della frequenza non è troppo elevato ma nemmeno troppo basso. Molte delle parole più frequenti all'interno dei documenti hanno infatti scarso valore dal punto di vista semantico: sono le *stop words*, ovvero articoli e preposizioni. La loro rimozione può portare ad una notevole riduzione della dimensione del dizionario ed al contempo consente di aumentare l'efficienza e l'efficacia del sistema di classificazione.
Il passo successivo consiste nell'eseguire operazioni di *stemming* sull'insieme di parole restanti, per ridurre le forme flesse delle parole alle loro radici: ad esempio termini come "user", "users", "used", "using" vengono tutti ricondotti alla stessa radice semantica "use". In questo modo si fanno corrispondere termini simili e le dimensioni del dizionario possono essere ulteriormente ridotte. I token così ottenuti vanno a costituire il vocabolario sulla cui base viene realizzato il modello dei documenti e che sarà utilizzato per la loro classificazione.
Per maggiori dettagli sulle diverse tecniche utilizzabili per la creazione del dizionario si rimanda allo studio condotto da Yang e Pedersen nel '97 [44].

2.1.2 Tipologie di rappresentazione

Data una collezione di documenti D, sia $K = \{k_1, k_2, \ldots, k_t\}$ il vocabolario stabilito a partire dai termini usati nei documenti. A ciascun documento può essere allora associato un vettore:

$$d_j = (w_{1j}, w_{2j}, \ldots, w_{tj}) \tag{2.1}$$

dove w_{ij} è il peso di ogni termine k_i nel documento d_j. Si avrà $w_{ij} > 0$ solo se il termine k_i è presente nel documento d_j. Nel seguito verrà impiegata la seguente terminologia:

- k_i indica un termine; d_j indica un documento; w_{ij} è il peso associato a (k_i, d_j);
- dato t il numero totale di termini, $K = (k_1, k_2, \ldots, k_t)$ è l'insieme dei termini;

- $vec(d_j) = (w_{1j}, w_{2j}, \ldots, w_{tj})$ si riferisce al vettore dei pesi associato al documento d_j;

Rappresentazione booleana

Si vuole valutare soltanto se un termine è presente o meno in un documento, ma non il numero di volte in cui compare. I pesi assumono pertanto dei valori booleani: $w_{ij} = \{0, 1\}$. In questo modello si avrà $w_{ij} = 1$ solo se il termine k_i è presente nel documento d_j.
Uno dei principali vantaggi dell'uso di questa tecnica risiede nel formalismo chiaro delle query, che sono indicate da espressioni booleane. Ad esempio, la query:

$$q = k_a \wedge (k_b \vee \neg k_c)$$

porta ai risultati mostrati in figura 2.2.

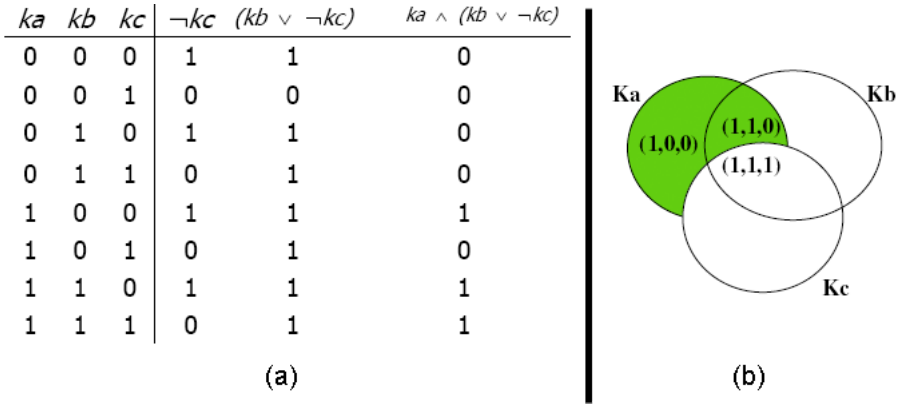


Figura 2.2. Modello booleano ($q = k_a \wedge (k_b \vee \neg k_c)$)

Spazio vettoriale

Il peso w_{ij} di ogni termine all'interno di un documento varia con continuità tra 0 e 1 in base al numero di occorrenze all'interno del testo. Ad ogni termine k_i viene associato un vettore di modulo unitario $vec(i)$, con solo un 1 alla posizione i-esima. Così si ottengono t versori che formano una

base ortonormale per uno spazio t-dimensionale. In questo spazio query e documenti sono rappresentati come vettori dei pesi:

$$vec(d_j) = (w_{1j}, w_{2j}, \ldots, w_{tj}) \qquad vec(q) = (w_{1q}, w_{2q}, \ldots, w_{tq})$$

Per il calcolo dei w_{ij}, sia per i documenti che per le query, vengono determinati per ogni k_i:

TF Term Frequency: il numero di occorrenze del termine k_i nel documento d_j, normalizzato rispetto alle frequenze di tutti gli altri termini nel documento d_j; indicando con f_{ij} la frequenza del termine k_i nel documento d_j, si ha:

$$TF_{ij} = \frac{f_{ij}}{max\{f_{1j}, f_{2j}, \ldots, f_{tj}\}} \tag{2.2}$$

IDF Inverse Document Frequency: il logaritmo dell'inverso della frequenza con la quale il termine k_i compare in tutti i documenti:

$$IDF_i = log\frac{N}{n_i} \tag{2.3}$$

con N pari al numero totale di documenti e n_i = numero di documenti nei quali compare k_i.

Il peso viene calcolato come:

$$w_{ij} = TF_{ij} \times IDF_i \tag{2.4}$$

Tale scelta è motivata dalla volontà di utilizzare una "buona" misura di peso, che tenga conto sia della similarità dei contenuti intra-document (TF) che del grado di dissimilarità inter-document (IDF).

2.1.3 Apprendimento di un modello

Data la collezione di documenti $D = \{d_1, d_2, \ldots, d_N\}$ e l'insieme delle categorie individuate dalle etichette $C = \{c_1, c_2, \ldots, c_L\}$, lo scopo della fase di apprendimento è approssimare la cosiddetta "funzione obiettivo" $\hat{\Phi} : D \times C \rightarrow \{T, F\}$, che descrive come un documento viene classificato, tramite

una $\Phi : D \times C \rightarrow \{T, F\}$, in modo che Φ e $\hat{\Phi}$ "coincidano il più possibile" [34].
A questo scopo l'insieme dei documenti a disposizione viene diviso nei due sottoinsiemi di *training* e *test*: il primo viene usato durante la fase di addestramento per la creazione del modello, il secondo per la verifica della sua validità.
Durante la fase di apprendimento viene calcolata una Φ sfruttando la conoscenza a priori della categorizzazione dei documenti del training set. La funzione così calcolata viene usata per classificare i documenti dell'insieme di test. In base ai risultati ottenuti è possibile stabilire se occorra modificare i valori di alcuni parametri del modello.
Uno dei metodi più usati per la stima dei parametri è la K-fold cross-validation: le tecniche di *K-fold cross-validation* prevedono il partizionamento del dataset in K sottoinsiemi aventi la stessa cardinalità, ciascuno dei quali viene utilizzato a turno come test set per la macchina addestrata sui rimanenti K-1 sottoinsiemi.

2.1.4 Metodi di classificazione

Data una collezione di documenti $D = \{d_1, d_2, \ldots, d_N\}$, un insieme di etichette $C = \{c_1, c_2, \ldots, c_L\}$ e una query q, l'obiettivo del processo di classificazione è assegnare un'etichetta al documento della query in base ad una funzione di classificazione Φ. Condizione necessaria a questo scopo è determinare una *funzione di similarità* $sim(q, d_j)$, definita come la "somiglianza" tra una query q ed un documento d_j. Esistono due tipologie di funzioni di questo genere:

- funzioni indicatrici: $sim(q, d_j) \in \{T, F\}$, permettono di stabilire solo se due documenti sono "simili" in modo assoluto;
- funzioni di rilevanza: $sim(q, d_j) \in [0, \sigma]$, $\sigma > 0$, forniscono una *misura di similarità* che consente di stabilire un ranking tra i documenti del modello.

Le funzioni del primo tipo possono fornire in modo diretto informazioni sulla classe di appartenenza del documento query, per esempio votando a maggioranza tra tutti gli elementi segnalati "simili"; una soluzione di questo tipo può però essere poco accurata, in quanto non fornisce una misura del grado di similarità. L'uso di funzioni di ranking consente invece di apprendere informazioni sul grado di similarità e di poter ordinare e raggruppare gli elementi in base a questi dati.

2.2 Bag of Visual Words

La ricerca di concetti di alto livello, capaci di rendere distinguibile un'immagine dalle altre in quanto esplicativi del suo contenuto semantico, si può tradurre tecnicamente nell'associazione tra immagine e *visual words* di un "vocabolario" che è possibile individuare al suo interno. Si tratta di un'applicazione della tecnica del Bag Of Words precedentemente introdotta per documenti testuali adattata ad un contesto visuale.
Anche in questo caso il processo prevede, prima della classificazione vera e propria, una fase di apprendimento su un insieme di immagini delle quali è nota la categoria di appartenenza; per ognuna di esse viene costruito un vettore i cui campi indicano la frequenza o, più semplicemente, la presenza delle parole del vocabolario (fig. 2.3). In questo modo i vettori relativi alle immagini di training vengono utilizzati per creare uno spazio delle categorie. Per classificare una nuova immagine basterà generarne il vettore di frequenza delle parole rispetto allo stesso vocabolario ed osservare la sua collocazione nello spazio delle categorie.
Le differenze principali rispetto alle applicazioni BoW per documenti testuali risiede nella creazione delle "parole" e in quella del "dizionario". Preso un insieme di immagini di training, si estraggono le informazioni di ciascuna di esse attraverso la ricerca di punti o regioni salienti. Le informazioni così ottenute vengono raggruppate per la generazione dei termini del vocabolario attraverso un'operazione di *clustering*, necessaria per ridurre la dimensionalità ad un valore fissato. Una volta ottenuto un vocabolario, ogni immagine viene descritta in base alla frequenza delle parole visuali al suo interno.

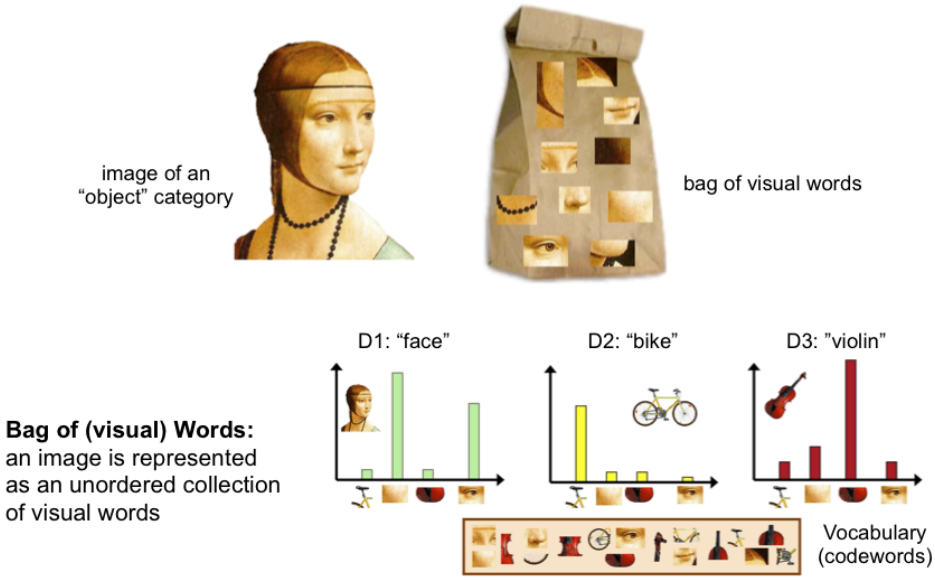


Figura 2.3. Esempio di creazione di modello BoW visuale

Il processo di creazione del *codebook* di parole visuali è l'analogo della creazione del dizionario nel caso testuale; l'analogia si estende anche alle operazioni di clustering e quantizzazione rispetto a quelle di stemming ed eliminazione delle stop-words. Mentre però nella classificazione di documenti testuali questi step sono opzionali e vengono introdotti solo al fine di migliorare le prestazione, il clustering e successiva quantizzazione dei feature vector è una condizione necessaria alla creazione del codebook visuale.
I passi principali della procedura di classificazione di immagini per mezzo del modello BoW sono i seguenti:

- **ricerca ed estrazione di feature**: ricerca e descrizione di punti salienti all'interno di ciascuna immagine, che concorreranno a formare il "dizionario visuale";

- **clustering-quantizzazione**: in analogia alle procedure di stemming per documenti testuali, vengono applicate delle tecniche di clustering dell'informazione all'insieme dei descrittori per ottenere un dizionario visuale delle dimensioni desiderate;

- **descrizione**: ogni immagine viene rappresentata per mezzo del *vettore delle frequenze* dei termini del dizionario al suo interno;
- **classificazione**: le nuove immagini vengono classificate in base ad appropriate misure di distanza.

2.3 Estrazione e descrizione delle feature

La ricerca di concetti all'interno di un'immagine ha come presupposto l'individuazione di informazioni di basso livello che ne possano descrivere il contenuto semantico. Il contributo informativo dell'immagine viene catturato dalla descrizione dei punti salienti (*keypoints*) che è possibile individuare al suo interno. Questi elementi di interesse si riferiscono a caratteristiche di tipo locale dell'immagine che offrano una certa robustezza ad alcune trasformazioni.
Lo schema generale seguito per la determinazione del contenuto informativo di un'immagine procede in due passi:

1. ricerca di caratteristiche puntuali o locali salienti;
2. descrizione delle caratteristiche trovate.

2.3.1 Ricerca di feature salienti

Le strategie di ricerca dei punti salienti possono essere distinte in due categorie: tecniche di campionamento sparso o denso. La prima categoria si avvale di metodi di detection basati sul contenuto dell'immagine, mentre la seconda mira a costruire un insieme di punti composto secondo criteri stabiliti a priori.
Il vantaggio principale che deriva dall'uso di una tecnica sparsa di ricerca di regioni salienti è che consente di determinare le zone a maggior contenuto informativo dell'immagine; d'altra parte però i risultati ottenuti con questo metodo dipendono in larga misura dall'algoritmo scelto e dal tipo/risoluzione dell'immagine, oltre a fornire spesso un numero basso di risultati. Una tecnica densa consente di descrivere il contenuto dell'immagine a livello globale e

può quindi rivelarsi utile in applicazioni BoW, ma fornisce poche informazioni sull'immagine in sé.

Campionamento sparso

Esistono molte tipologie diverse di algoritmi di ricerca di feature locali salienti in immagini: ad esempio [25] offre un'ampia descrizione ed un confronto tra i principali. Tra di essi uno di quelli che offre le migliori performance è senza dubbio **SIFT (Scale Invariant Feature Transform)**, introdotto da Lowe in [22]: si tratta di un metodo per la rilevazione e la descrizione di caratteristiche locali di un'immagine attraverso la ricerca di keypoints. SIFT offre buone performance ed è invariante a rotazione, scala, variazioni di intensità e moderate trasformazioni affini.
I punti salienti individuati da SIFT godono della proprietà di essere massimi locali all'interno dello scale-space di Differenze di Gaussiane (DoG); essi corrispondono a punti dell'immagine che rimangono stabili a scale diverse. Una rappresentazione piramidale dell'immagine *a scala di grigi* è ottenuta per mezzo di convoluzioni successive con filtri gaussiani a scale diverse (vedi immagine 2.4):

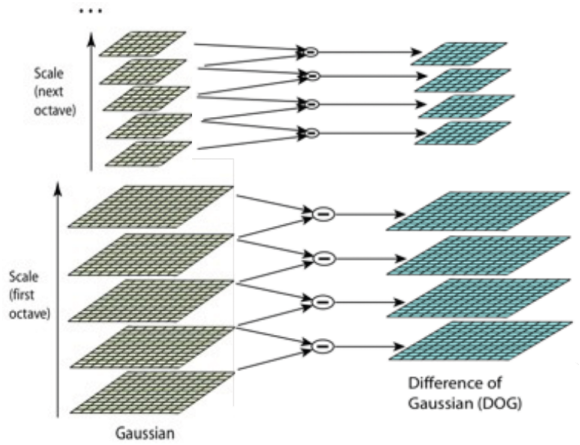


Figura 2.4. Rappresentazione piramidale di un'immagine nello scale-space DoG

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2.5}$$

dove $I(x,y)$ è l'intensità del pixel (x,y) e $G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}}$. Le DoG sono ottenute come differenze tra livelli adiacenti della piramide:

$$D(x,y,\sigma) = L(x,y,k\sigma) - L(x,y,\sigma) \tag{2.6}$$

Per ogni livello di DoG ciascun pixel viene confrontato con i suoi 8 vicini alla stessa scala e con i 9 vicini individuati sulle scale adiacenti (inferiore e superiore). Soltanto i pixel che risultano essere estremi locali (massimi o minimi) in questo insieme vengono scelti come keypoints. Il valore della scala σ all'interno della piramide DoG a cui viene individuato il keypoint viene indicato come *scala caratteristica* del punto.

Campionamento denso

Soluzioni di ricerca di punti salienti alternative prevedono di descrivere ogni immagine secondo un numero fisso di punti. Tali tecniche sono dette di campionamento denso, in particolare si riferiscono a:

1. campionamento a griglia (Regular Grid): fissato un passo di campionamento, i punti descritti corrispondono ai nodi di una griglia sovrapposta all'immagine;

2. campionamento casuale (Random): un numero prefissato di punti è scelto in modo casuale tra tutti i pixel dell'immagine.

La figura 2.5 mette a confronto le diverse tecniche di campionamento. Usando

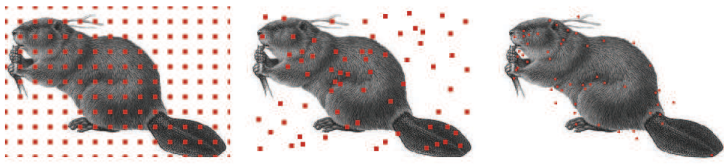


Figura 2.5. Da sinistra: Regular Grid (passo 10), Random (100 punti), regular SIFT

una tecnica di campionamento denso non ha senso parlare di scala caratteristica assegnata ad un punto. Tale valore viene impostato a $\sigma = 1$ in caso

si desideri ottenere una descrizione dell'immagine così com'è; considerando valori in insiemi più vasti, ad esempio $\sigma \in \{0.5, 1, 1.5, 2\}$, equivale a utilizzare un numero di punti quadruplo rispetto al caso precedente, ognuno dei quali verrà descritto ad una scala diversa.
Esperimenti [11] hanno mostrato che per applicazioni di classificazione basate su bag of words di scene naturali le tecniche di campionamento denso basato su una griglia regolare offrono ottime prestazioni.

2.3.2 Descrittore SIFT

Fissata una scala caratteristica per un keypoint, il calcolo del descrittore SIFT si articola in due fasi successive:

1. calcolo dell'orientazione canonica del keypoint;

2. calcolo del vettore di feature.

Prima di tutto si determina l'immagine $L(x, y, \sigma)$ alla scala corrispondente a quella assegnata al keypoint. In questa immagine si accumulano in un istogramma a 8 dimensioni le direzioni del gradiente in una finestra di 4x4 pixel intorno al keypoint. Il picco registrato nell'istogramma corrisponde alla *direzione dominante* del gradiente locale per il punto corrente. La determinazione dell'orientazione canonica rende il descrittore invariante a rotazioni. Il descrittore viene calcolato considerando una finestra di 16x16 locazioni nell'intorno del keypoint alla scala ad esso essegnata, prendendo come sistema di riferimento quello individuato dall'orientazione canonica. Tali locazioni vengono raggruppate in sottoregioni 4x4, per ciascuna della quali è calcolato un istogramma a 8 dimensioni delle orientazioni del gradiente locale, sempre in riferimento alla direzione dominante. I valori di ogni bin dell'istogramma equivalgono alla somma delle ampiezze dei vettori gradiente con direzione nel range del bin considerato. Il risultato finale è un vettore a 128 dimensioni: 16 (4x4) x 8. (vedi figura 2.6) Il descrittore del keypoint è normalizzato per renderlo invariante a cambi di intensità

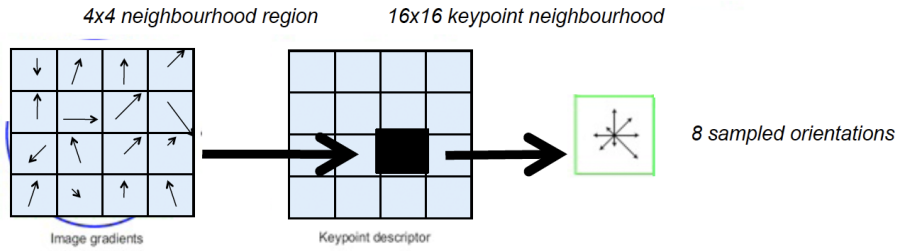


Figura 2.6. Descrittore SIFT

2.3.3 Descrittore ColorSIFT

In [39], Van de Sande *et al.* presentano uno studio delle proprietà di invarianza e del grado di distintività di descrittori basati su SIFT ed estesi alla rappresentazione di informazioni di colore. Le diverse soluzioni si differenziano in base allo spazio di colore utilizzato per la rappresentazione dell'immagine ed al tipo di approccio utilizzato per la descrizione SIFT. L'autore ha reso disponibile un software per la descrizione di immagini secondo un modello di campionamento denso che consente di utilizzare sia la versione a scale di grigio "classica" di Lowe che le possibili varianti che impiegano l'immagine a colori.

HSV-SIFT: il descrittore SIFT viene calcolato sui tre canali dello spazio di colore HSV (Hue Saturation Value, fig. 2.7), che corrispondono singolarmente ad un'immagine a scala di grigi dell'implementazione originaria. Il risultato è un descrittore di dimensione complessiva $3 \times 128 = 384$; questa soluzione è stata utilizzata ad esempio in [6].

HueSIFT: introdotto in [40], deriva dalla concatenazione dell'istogramma di colore HSV normalizzato rispetto alla tonalità con il corrispondente descrittore SIFT, calcolato su patch locali (fig. 2.8); dato che lo hue histogram viene rappresentato su 37 bin, la dimensione finale del descrittore è 165.

OpponentSIFT: Descrive tutti i canali dello spazio di colore *opponent* con

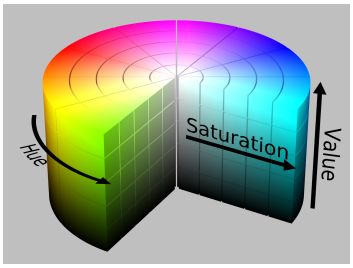


Figura 2.7. Sistema di coordinate cilindriche HSV

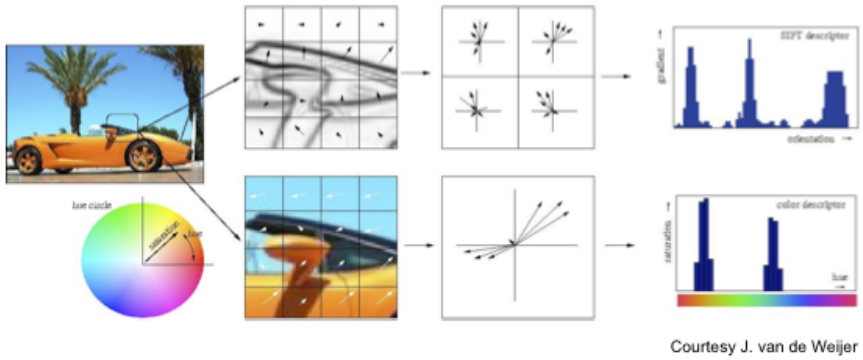


Figura 2.8. Creazione del descrittore HueSIFT

SIFT:

$$\begin{pmatrix} O_1 \\ O_2 \\ O_3 \end{pmatrix} = \begin{pmatrix} \frac{R-G}{\sqrt{2}} \\ \frac{R+G-2B}{\sqrt{6}} \\ \frac{R+G+B}{\sqrt{3}} \end{pmatrix}$$

Il canale O_3 fornisce informazioni sulla luminanza, O_1 ed O_2 sul colore. Anche in questo caso si ottiene un descrittore finale di dimensione $3 \times 128 = 384$ per ogni keypoint.

C-SIFT: dato che i canali O_1 e O_2 dello spazio di colore opponent contengono comunque informazioni di luminanza, viene introdotto anche un descrittore ad "invariante di colore" C-SIFT che consiste nel normalizzare i due canali rispetto alla terza componente: O1/O3, O2/O3; il descrittore risultante ha le stesse dimensioni di OpponentSIFT (384).

rgSIFT: Il modello RGB normalizzato viene calcolato come segue:

$$\begin{pmatrix} r \\ g \\ b \end{pmatrix} = \begin{pmatrix} \frac{R}{R+G+B} \\ \frac{G}{R+G+B} \\ \frac{B}{R+G+B} \end{pmatrix}$$

Nel descrittore rgSIFT vengono aggiunti al SIFT classico i descrittori per le componenti cromatiche r e g di tale modello (dim. 384).

tcSIFT: la *transformed color distribution* deriva da RGB ma a differenza di questo risulta invariante a modifiche di illuminazione:

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = \begin{pmatrix} \frac{R-\mu_R}{\sigma_R} \\ \frac{G-\mu_G}{\sigma_G} \\ \frac{B-\mu_B}{\sigma_B} \end{pmatrix}$$

dove μ_C e σ_C rappresentano media e deviazione standard della distribuzione nel canale C calcolate nell'area di interesse per il descrittore (patch locale). Il descrittore SIFT viene applicato ai tre canali RGB normalizzati così ottenuti (dim. 384).

RGB-SIFT: descrittore SIFT applicato ad ognuno dei tre canali R, G e B separatamente (dim. 384).

2.4 Creazione del vocabolario

La costruzione del modello bag of words implica la creazione di un *codebook*, il vocabolario visuale discreto. Un vocabolario nel dominio della classificazione di oggetti/scene può essere calcolato seguendo due approcci:

- annotazione: il vocabolario è ottenuto assegnando etichette significative a porzioni dell'immagine (es. cielo, acqua, vegetazione, ...);
- *data-driven*: i termini del vocabolario sono calcolati come centroidi ottenuti al termine di un processo di clustering delle feature.

In tecniche di creazione del codebook data-driven è necessario eseguire una *quantizzazione* di un elevato numero di vettori di feature, rappresentati in genere in uno spazio a elevata dimensionalità; per fare questo si utilizzano tecniche di clustering che consentono di definire le parole visuali. Ad esempio in figura 2.9 viene mostrato un semplice esempio in cui si identificano due parole visuali a partire dall'insieme delle patch individuate nelle immagini. Le

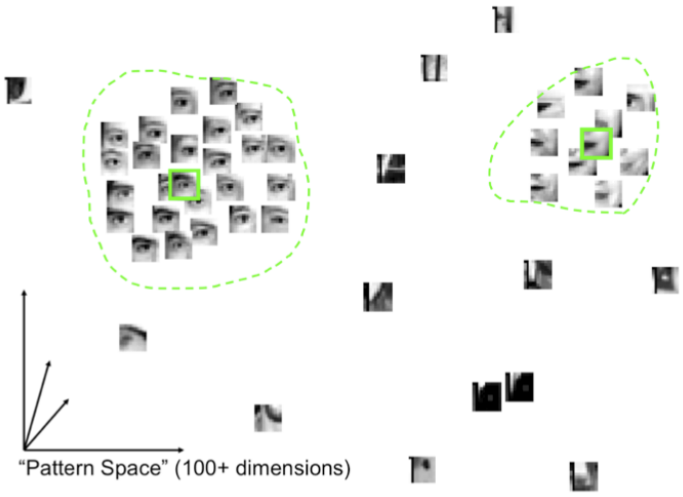


Figura 2.9. Creazione di parole visuali tramite clustering

performance di questo metodo dipendono dal metodo di quantizzazione scelto e dal numero di parole visuali che compongono il codebook. L'approccio di quantizzazione più comunemente usato è *K-means*, vista la sua relativa semplicità e la velocità di convergenza.

2.4.1 Algoritmo di clustering K-means

L'algoritmo K-Means è un algoritmo di clustering non supervisionato che permette di suddividere gruppi di oggetti in K partizioni sulla base dei loro attributi. Si assume che tali attributi possano essere rappresentati come vettori, e che quindi formino uno spazio vettoriale.
Siano dati N oggetti con i attributi, in uno spazio vettoriale i-dimensionale,

definiamo:

$$X = X_1, X_2, \ldots, X_N$$

come insieme degli oggetti su cui eseguire il clustering.
Il nostro scopo è ottenere un insieme di cluster:

$$P = P_1, P_2, \ldots, P_K \quad 1 \ll K \ll N$$

tali che:

- $\bigcup_1^K P_i = X$: tutti gli oggetti devono appartenere ad almeno un cluster;
- $\bigcap_1^K P_i = \emptyset$: ogni oggetto può appartenere ad un solo cluster;
- $\emptyset \subset P_i \subset X$: nessun cluster può essere vuoto o contenere tutti gli oggetti.

La partizione viene indicata con una matrice $U \in \mathrm{N}^{KxN}$, il cui generico elemento $u_{ij} = \{0, 1\}$ indica l'appartenenza dell'oggetto j al cluster i. Indicato con

$$C = C_1, C_2, \ldots, C_K$$

l'insieme dei K centroidi, uno per ciascun cluster, l'obiettivo è minimizzare la funzione errore:

$$V(U, C) = \sum_{i=1}^{K} \sum_{X_j \in P_i} \|X_j - C_i\|^2 \tag{2.7}$$

In generale, il clustering k-means è un problema NP-hard, quindi sono stati

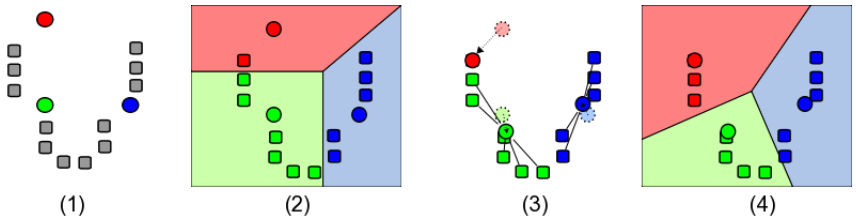


Figura 2.10. Esempio di esecuzione di K-means (K=3)

sviluppati una serie di algoritmi euristici per il suo calcolo. Il più comunemente usato è detto **algoritmo di Lloyd**.

Dato un insieme iniziale di K centroidi, $C_1^{(1)}, \ldots, C_K^{(1)}$, scelti a caso in base al dataset fornito (fig.2.10(1)), l'algoritmo procede alternando due passi di elaborazione:

Assegnazione ciascuno degli elementi da esaminare viene assegnato al cluster il cui centroide risulta più vicino (fig.2.10(2)):

$$P_i^{(t)} = \left\{ X_j \ : \ \|X_j - C_i^{(t)}\| \leq \|X_j - C_{i*}^{(t)}\| \ \forall i* = 1, \ldots, K \right\} \tag{2.8}$$

Aggiornamento Calcolo dei nuovi centroidi associati a ciascun cluster (fig.2.10(3)):

$$C_i^{(t+1)} = \frac{1}{|P_i^{(t)}|} \sum_{X_j \in P_i^{(t)}} X_j \tag{2.9}$$

Si dimostra che l'algoritmo converge quando l'assegnazione non cambia più (fig.2.10(4)).

Essendo un algoritmo euristico, non c'è garanzia che esso converga ad un ottimo globale ed il risultato finale può dipendere dalla scelta dei cluster iniziali. Dato che l'algoritmo è in genere molto veloce, si tende a ripetere la sua esecuzione più volte con differenti condizioni di partenza. È stato dimostrato che esistono certi insiemi di punti per i quali k-means converge in tempo superpolinomiale: $2^{\Omega(\sqrt{N})}$.

2.4.2 Creazione di dizionari testuali e visuali

Per creare un buon vocabolario di termini a partire da un insieme di documenti testuali si devono prendere in considerazione due fattori principali: la selezione delle feature e la dimensione del vocabolario. È necessario utilizzare solo termini che siano realmente rilevanti dal punto di vista semantico, basandosi ad esempio su punteggi calcolati utilizzando statistiche adeguate (infogain, χ^2) e su tecniche di filtraggio come quelle già illustrate nel paragrafo 2.1.1. L'applicazione di queste tecniche offre buoni risultati perché, come è già stato descritto in precedenza, la distribuzione dei termini in un corpus segue la regola di Zipf.

È interessante provare ad applicare tecniche simili alle parole visuali in un

corpus di immagini, per stabilire se la regola di Zipf risulta ancora valida e se quindi sia consigliabile applicare strategie analoghe al caso testuale per la ricerca dei termini di maggior rilevanza. Yang et al. [42] riportano un esperimento in cui rimuovere le parole più frequenti porta ad una diminuzione costante nelle prestazioni di classificazione su due dataset di riferimento: PASCAL[1] e TRECVID[2]. Nella stessa fonte viene inoltre mostrato che l'uso di *tf-idf* invece del solo *tf* non porta sempre a risultati migliori; i risultati ottenuti sono presentati in figura 2.11.

Corpus	**Vocabulary size**	Linear SVM					RBF SVM				
		bxx	*txx*	*txc*	*tfx*	*tfc*	*bxx*	*txx*	*txc*	*tfx*	*tfc*
TRECVID	200	0.095	**0.152**	0.109	0.147	0.110	0.137	**0.167**	0.112	0.130	0.108
	1,000	0.139	0.162	0.137	**0.183**	0.142	**0.235**	0.202	0.141	0.161	0.128
	5,000	0.183	0.178	0.150	**0.205**	0.153	**0.245**	0.224	0.141	0.194	0.145
	20,000	**0.237**	**0.228**	0.185	0.225	0.188	**0.271**	**0.278**	0.163	0.216	0.184
PASCAL	200	0.465	0.680	0.605	0.639	**0.693**	0.513	0.670	**0.742**	0.619	0.686
	1,000	**0.681**	0.677	0.677	**0.690**	**0.683**	**0.754**	0.639	0.751	0.618	0.722
	5,000	**0.764**	0.738	0.745	0.740	0.745	**0.777**	0.708	0.737	0.757	0.734
	20,000	**0.721**	0.682	0.708	0.682	0.711	**0.683**	0.642	**0.690**	0.528	**0.682**

* Weighting: *bxx = binary, txx = tf, txc = tf + normalization, tfx = tf + idf, tfc = tf + idf + normalization*

Figura 2.11. Performance di classificazione (MAP) su TRECVID e PASCAL al variare degli schemi di pesatura e delle dimensioni del vocabolario

2.5 Descrizione

In un modello di tipo bag of words puro applicato a documenti testuali non si tiene conto dell'ordine in cui compaiono le parole: il documento viene descritto tramite il vettore delle occorrenze dei termini del dizionario all'interno di esso. Analogamente nel caso visuale ogni punto saliente individuato nell'immagine viene assegnato al centro del cluster ad esso più vicino in base al metodo scelto in fase di creazione del codebook. Proprio grazie a questo processo non si parla più di punti salienti ma di *visual words*. Il descrittore dell'immagine è un istogramma di dimensione pari alla cardinalità del codebook, in cui ogni elemento mostra la frequenza della parola visuale cor-

[1]http://pascallin.ecs.soton.ac.uk/challenges/VOC/

[2]http://www-nlpir.nist.gov/pro jects/trecvid/

rispondente all'interno dell'immagine, cioè quante volte un punto saliente è stato etichettato con il cluster associato alla parola. Un esempio di creazione di un vettore di parole visuali di questo tipo è presentato in figura 2.12.

Come si intuisce la posizione relativa dei punti salienti non viene presa in

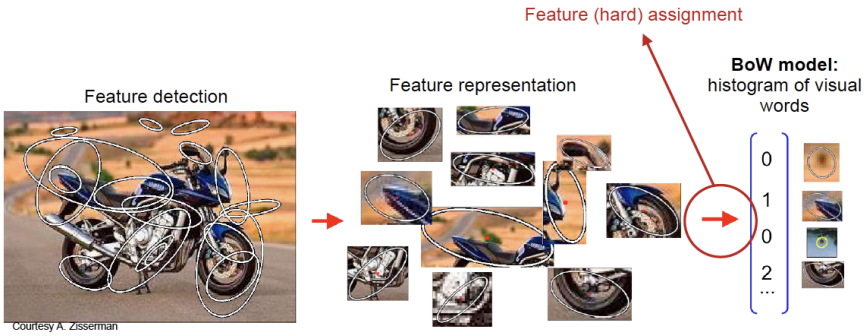


Figura 2.12. Assegnamento delle feature alle parole ottenute tramite quantizzazione

considerazione, in accordo con la tipologia di rappresentazione tramite bag of words. Questa caratteristica semplifica di molto la costruzione del modello delle parole visuali ma al contempo può costituire una delle principali debolezze del metodo: è evidente che gli oggetti complessi rappresentati nelle immagini sono costituiti da parti che si trovano in specifiche relazioni spaziali tra di essi, caratteristiche per il tipo di oggetto. Si consideri ad esempio il caso di un volto, come rappresentato in figura 2.13. Ipotizzando che siano state individuate parole visuali corrispondenti a "occhio destro", "occhio sinistro", "naso", "bocca", in un modello BoW tutte le configurazioni spaziali presentate in figura verrebbero considerate come equiprobabili, mentre solo l'ultima identifica realmente un volto e sarebbe da considerare di maggior rilevanza semantica.

Piramide spaziale

Viste le considerazioni fatte al termine del precedente paragrafo sui possibili svantaggi derivanti dall'uso del bag of words, esistono soluzioni da esso de-

Figura 2.13. Le tre configurazioni spaziali sono tutte equiprobabili in un modello BoW, ma solo la terza è realistica

rivate che utilizzano informazioni di tipo spaziale. Uno di essi è il metodo descritto da Lazebnik *et al.* in [37], che propone un sistema di riconoscimento delle scene basato su criteri di corrispondenza geometrica globale. La tecnica adottata consiste nel partizionamento ricorsivo dell'immagine in regioni di dimensioni sempre più piccole, per ciascuna delle quali vengono via via calcolati gli istogrammi delle feature locali. La descrizione finale dell'immagine deriva dalla concatenazione degli istogrammi calcolati alle diverse risoluzioni, associando ad ognuno un peso opportuno in modo da dare maggiore rilevanza alle corrispondenze individuate a risoluzione più fine.
La rappresentazione che si ottiene viene detta "piramide spaziale", un'estensione della descrizione "senza ordine" offerta dal metodo classico di bag of words. Una rappresentazione classica di tipo BoW, secondo gli autori, offre infatti un limitato potere descrittivo per la mancanza di informazioni che descrivano il layout spaziale delle feature. Una descrizione di tipo piramidale, invece, offre buone prestazioni dal punto di vista computazione e migliori risultati per quanto riguarda le operazioni di classificazione.
Questo tipo di descrizione verrà affrontata in maniera diffusa nel paragrafo 4.2.3.

2.6 Classificazione

Come nel caso di categorizzazione di documenti testuali, per classificare immagini di test è necessario costruire un *modello matematico* che consenta

di rappresentare formalmente le proprietà delle immagini usate per l'addestramento. Sono necessarie dunque tecniche di apprendimento automatico che a partire dalle descrizioni ottenute sulle immagini note possano costruire un modello di classificazione che consenta di analizzarne di nuove. Questo processo viene generalmente compiuto usando tecniche di apprendimento *supervisionato* o *semi-supervisionato*: i dati di addestramento sono rappresentati come coppie di elementi $\{(x_i, c_i)\ i = 1, \ldots N\}$, dove con x_i si indica un generico vettore descrittore del dato i-esimo (ad esempio l'istogramma BoW dell'immagine) e con y_i la classe a cui appartiene tale elemento. Il risultato del processo di apprendimento a partire da questi dati porta alla formulazione di una funzione di classificazione $f : X \rightarrow C = \{c_1, c_2, \ldots, c_C\}$ che consente di associare una classe c_q ad ogni immagine di test x_q.

2.6.1 Metodi Nearest Neighbor

L'algoritmo *k nearest neighbors (k-NN)* è una delle soluzioni più semplici di apprendimento automatico: a un oggetto viene assegnata la classe più frequente tra i suoi k vicini (fig. 2.14). Se $k = 1$, all'oggetto viene assegnata la classe dell'elemento ad esso più vicino. Questo algoritmo può essere definito di tipo semi-supervisionato in quanto non fa uso delle etichette assegnate agli oggetti durante la ricerca dei vicini, ma solo nell'ultima fase di votazione.

La fase di addestramento consiste nel semplice salvataggio dei vettori di

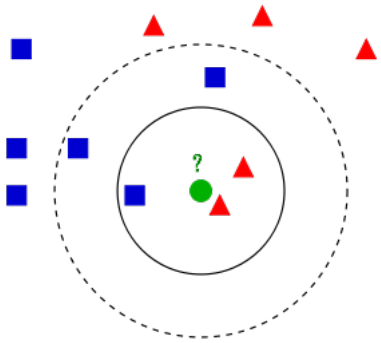


Figura 2.14. Esempio di classificazione K-NN di un elemento non etichettato (in verde): ponendo $k = 3$ (cerchio continuo), la classe assegnata sarà quella dei triangoli rossi; con $k = 5$ quella dei quadrati blu.

feature e delle etichette per tutti gli elementi dell'insieme di training. La classificazione di un vettore di feature non etichettato avviene cercando la classe che appare più spesso nell'insieme dei suoi k vicini. Per stimare la "vicinanza" tra due elementi è necessario stabilire quale criterio di distanza impiegare. In genere viene impiegata la distanza euclidea, detta anche *distanza L2* in quanto pari alla norma-2 del vettore differenza:

$$d_{L2}(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} = ||x - y|| \tag{2.10}$$

Una formula alternativa per il calcolo della distanza è la cosiddetta χ^2 (*chi-square*), una forma particolare di distanza L2 quadratica pesata: in essa gli elementi del vettore devono rappresentare delle frequenze relative e le differenze quadratiche tra gli elementi sono pesate da un parametro associato al loro valore medio. In formule:

$$\chi^2(\overline{x}, \overline{y}) = \sum_{i=1}^{n} \frac{1}{2(x_i + y_i)} \cdot (x_i - y_i)^2 \tag{2.11}$$

$$\overline{x} = [\overline{x}_1 \dots \overline{x}_n] \quad \overline{x}_i = \frac{x_i}{\sum_{k=1}^{n} x_k} = \frac{x_i}{||x||_1}$$

con $\overline{x}$ ad indicare i vettori normalizzati secondo la norma-1.
La scelta del valore di k dipende dai dati a disposizione per l'addestramento ed il test: valori più grandi riducono l'effetto del rumore nella classificazione, ma rendono i confini tra le classi meno distinti. Un buon valore di k può essere scelto eseguendo più prove modificando i dati di train-test, ad esempio usando una k-fold cross-validation.

2.6.2 Support Vector Machines

Le SVM (Support Vector Machine) sono state sviluppate negli AT&T Bell Laboratories principalmente da Vladimir Naumovich Vapnik. Nate per applicazioni di OCR (Optical Character Recognition), impiegano tecniche di *apprendimento supervisionato* per risolvere problemi di classificazione e regressione. Vista come classificatore binario, una SVM ha lo scopo di individuare il "confine" tra punti appartenenti a due classi: i nuovi punti saranno

classificati in base alla loro posizione rispetto a questo iperpiano di separazione, che massimizza la distanza dagli esempi di training più vicini. Le SVM hanno alcune interessanti proprietà:

- overfitting altamente improbabile;
- possibilità di gestione di dati multidimensionali e classificazione multiclasse;
- possibilità di individuare un sottoinsieme di esempi di training effettivamente necessari alla classificazione, detti vettori di supporto.

Le SVM implementano una tecnica di apprendimento supervisionato per la classificazione multiclasse. Per semplificare l'esposizione, di seguito sarà fornita una descrizione del funzionamento di SVM per un problema di classificazione binaria; nella pratica questa metodologia viene in genere impiegata anche per la classificazione multiclasse, applicando una tecnica di tipo one-vs-all che prevede di considerare ogni volta una categoria come positiva, tutte le altre come negative.
Sia dato un insieme di esempi già classificati $\{(x_i, y_i)\ i = 1, \dots N\}$, con $x_i \in \mathbb{R}^N$ e $y_i \in \{-1, +1\}$. Obiettivo dell'apprendimento è determinare la probabilità $P(Y|X = x)$ che un esempio appartenga ad una classe, ovvero individuare l'iperpiano di separazione ottimo tra i due insiemi di esempi:

$$f_C : \mathbb{R}^N \rightarrow \{+1, -1\} \ \mid \ x_i \rightarrow y_i = \beta^T x_i + \beta_0 \tag{2.12}$$

Posto in questi termini, l'obiettivo diventa la ricerca dell'iperpiano a massimo margine (*Maximal Margin Hyperplane - MMH*) tra due insiemi di punti nello spazio $\mathbb{R}^N$, che può essere generalizzato, per punti non linearmente separabili, in un *Support Vector Classifier (C-SVC)*, mostrato in figura 2.15.
Il problema di ottimizzazione può essere formalizzato come segue:

$$(P) \quad \min_{\beta, \beta_0, \xi} \quad \frac{1}{2}\|\beta\|^2 + C \sum_{i=1}^{N} \xi_i \tag{2.13}$$

con C detto *parametro di costo*. Detto più semplicemente, la risoluzione del problema (P) consente di trovare l'iperpiano a massimo margine che minimizza il numero di punti classificati erroneamente. Il valore ottimo trovato

ha la forma seguente:

$$\widehat{f}_C(x) = \hat{\beta}^T x + \beta_0 = \sum_{i \in SV} \hat{\alpha}_i y_i \left(x_i^T x\right) + \beta_0 \tag{2.14}$$

con SV insieme dei *vettori di supporto*, ovvero insieme degli esempi che si trovano a distanza M dall'iperpiano di separazione o che sono erroneamente classificati (fig. 2.15). Si dimostra la funzione così ottenuta minimizza l'errore empirico:

$$E(C) = \frac{1}{n} \sum_{i=1}^{n} |f_C(x_i) - y_i| \tag{2.15}$$

Analizzando la formula 2.14 si possono trarre due conclusioni rilevanti:

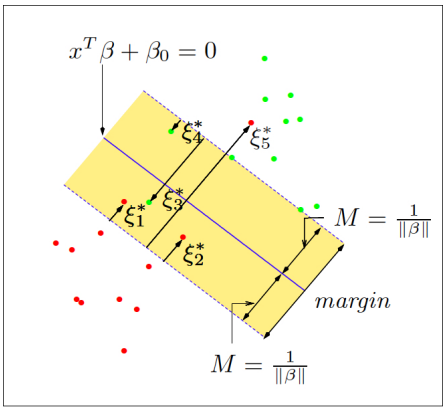


Figura 2.15. Support Vector Classifier

1. il problema di ricerca del MMH, ovvero il problema di classificazione, può essere ridotto alla ricerca dei pesi α_i da assegnare a ciascun vettore di supporto $x_i, i \in SV$;

2. gli esempi x_i appaiono sempre sotto forma di prodotto scalare.

Introducendo una funzione di feature mapping $\phi : \ x \rightarrow \phi(x)$ possiamo mappare i dati iniziali in uno spazio di dimensione superiore (anche infinita, in caso di *spazi di Hilbert*), nel quale i dati siano linearmente separabili. La formula per il calcolo dell'iperpiano di separazione diventa:

$$\widehat{f}(x) = \sum_{i \in SV} \hat{\alpha}_i y_i \langle \phi(x_i), \phi(x) \rangle + \beta_0 \tag{2.16}$$

Viste le considerazioni fatte sulla formula dell'iperpiano, possiamo notare che non c'è bisogno della conoscenza esplicita della funzione ϕ, ma solo della corrispondente *funzione kernel*:

$$k(x_i, x) = \langle \phi(x_i), \phi(x) \rangle \tag{2.17}$$

Per mezzo del cosiddetto *kernel trick*, dunque, la funzione di separazione ottima assume la forma:

$$\widehat{f}(x) = \sum_{i \in SV} \hat{\alpha}_i y_i k(x_i, x) + \beta_0 \tag{2.18}$$

Una funzione $k(x_i, x)$ è un kernel valido se esiste una funzione di *feature mapping* ϕ che soddisfa la relazione 2.17. Considerando la matrice Gram $K: \ K_{i,j} = k(x_i, x_j)$, k è un kernel valido se e solo se K è simmetrica e PSD (Semi-definita positiva, condizioni di Mercer), ovvero se K è simmetrica e tutti i suoi autovalori sono positivi.

Capitolo 3

Modelli a topic latenti

3.1 Dal modello bag of words al modello generativo

Il capitolo precedente ha fornito un'ampia panoramica dell'uso dello schema bag of words per la descrizione e classificazione di collezioni di documenti testuali o visuali. Il conteggio delle frequenze con cui una parola codificata in un vocabolario precalcolato appare nei documenti osservati è uno dei criteri più in uso nelle tecniche di Information Retrieval (IR) e viene spesso formalizzato attraverso la definizione del **tf-idf**: scelto un vocabolario di "parole" V, il term frequency **tf** indica il numero di occorrenze di ogni parola per ciascun documento del corpus, dove per "corpus" si intende un insieme di documenti M; l'inverse document frequency **idf** misura invece la frequenza di ogni termine su tutto il set di documenti. Il risultato finale del calcolo delle frequenze è una matrice termine-documento $X(V \times M)$ in cui le cui colonne contengono i valori di tf-idf per ogni documento del corpus e ciascuna riga è relativa ad un termine del vocabolario.
Lo schema tf-idf ha il pregio di individuare insiemi di parole che siano discriminanti per i documenti del corpus ma offre una limitata riduzione della descrizione degli stessi e non permette di inferire una struttura statistica intra/inter classe sulla collezione [5]. Uno dei metodi proposti per superare questa limitazione è ad esempio il **latent semantic indexing (LSI)**: LSI

usa una decomposizione a valori singolari della matrice X per identificare un sottospazio lineare di feature tf-idf che catturino la gran parte della varianza della collezione: questa procedura, applicata alla descrizione di documenti testuali, consente di ricavare informazioni di tipo linguistico come sinonimia e polisemia. Anche se l'uso di LSI consente di ottenere una consistente compressione per larghe collezioni di documenti, rimane comunque aperto il problema catturare le statistiche inter- ed intra-documento. È necessario utilizzare quindi un qualche modello di tipo probabilistico.

Un **modello probabilistico generativo** si basa sull'assunto che i dati osservabili di un insieme siano generati da un qualche processo casuale parametrizzato. Calcolato il set di parametri che meglio si adatta ai dati a disposizione, si può utilizzare il modello per predire o simulare valori per tutte le le variabili del modello. Questa è una delle differenze principali che distinguono i modelli generativi da quelli discriminativi, come ad esempio SVM, per i quali i valori assegnabili ad una variabile possono essere unicamente campionati sulle quantità osservabili nel set di dati in input; i modelli generativi inoltre possono catturare relazioni molto complesse tra le osservazioni e le variabili del modello.

Nei paragrafi seguenti verrà proposta una descrizione dei principali modelli generativi utilizzati nell'analisi di immagini e di testi, ovvero pLSA (*probabilistic Latent Semantic Analysis*) ed LDA (*Latent Dirichlet Allocation*), preceduta da una breve introduzione delle notazioni e delle nozioni necessarie ad una prima comprensione. Prima di passare alla descrizione formale vera e propria è possibile anticipare alcuni concetti di base propedeutici alla comprensione del modo in cui pLSA ed LDA si adattano alla descrizione e classificazione di documenti.

I modelli generativi affermano che ogni parola in un documento è campionata a partire da un modello a mistura[1], le cui componenti sono variabili casuali multinomiali che possono essere viste come rappresentazione dei "to-

[1]In statistica un modello a mistura è un modello probabilistico per la rappresentazione di sotto-popolazioni all'interno di una popolazione generale; tale rappresentazione non richiede di disporre di un dataset di osservazioni che identifichi la sotto-popolazione a cui una data osservazione appartiene.

pic", ovvero le categorie semantiche o "argomenti" da cui le parole vengono generate. Ogni parola è generata da un topic e ciascun documento contiene parole generate da differenti topic. Ogni documento è quindi composto da un insieme di proporzioni relative alle componenti della mistura ed in definitiva può essere espresso in termini di distribuzione di probabilità su un numero prefissato di topic: questa distribuzione corrisponde alla descrizione associata al documento all'interno del dataset.

3.1.1 Modelli grafici

La rappresentazione formale dei modelli generativi in esame fa uso di una notazione basata sui modelli grafici, pertanto risulta utile fornire in questo paragrafo una descrizione dei simboli utilizzati in questo formalismo. La figura 3.1 mostra la rappresentazione dei componenti che formano un modello grafico.

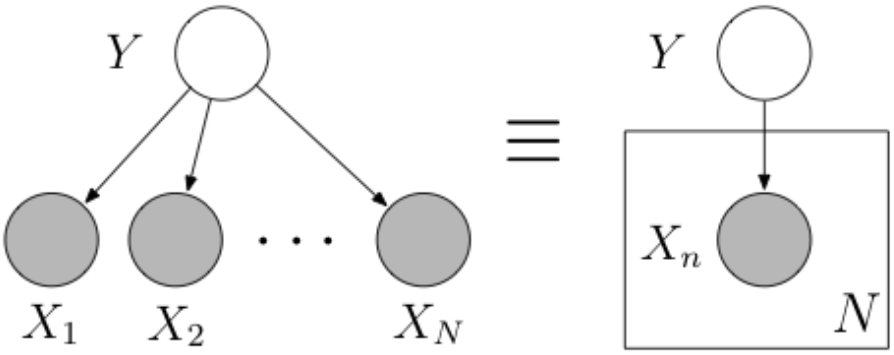


Figura 3.1. Componenti di un modello grafico

I simboli raffigurati nella figura hanno il seguente significato:

- i nodi rappresentano variabili aleatorie;
- gli archi denotano una possibile dipendenza;
- i nodi scuri si riferiscono a variabili osservate;

- i rettangoli indicano strutture replicate;
- la struttura del grafo definisce le relazioni di dipendenza condizionata tra le variabili casuali.

In termini di distribuzione di probabilità il grafico appena visto equivale alla formula:

$$p(y, x_1, \dots x_N) = p(y) \prod_{n=1}^{N} p(x_n|y)$$

dove le x_n sono tutte condizionatamente indipendenti data y.

3.1.2 Notazioni e terminologia

Anche se i modelli generativi in esame, ad esempio LDA, sono stati utilizzati inizialmente per descrivere collezioni di testi, i principi sui cui poggiano possono essere applicati in generale a qualsiasi problema che riguardi la classificazione di collezioni di dati. Di seguito viene riportata la definizione formale di alcuni termini utilizzati nel paragrafo introduttivo quali "parola", "documento" e "corpus" per indicare le entità coinvolte nel prosieguo della trattazione.
Formalmente:

- "parola": costituisce l'unità di base di dati discreti, definita come un elemento di un vocabolario di V termini. Le parole vengono rappresentate con vettori che hanno un singolo componente ad uno e gli altri a zero: la v-esima parola del vocabolario viene rappresentata da un V-vettore w tale che $w^v = 1$ e $w^u = 0$ per ogni $u \neq v$.
- "documento": una sequenza di N parole: $\mathbf{w} = (w_1, w_2, \dots, w_N)$.
- "corpus": una collezione $\mathbf{D} = (\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_M)$ di M documenti.
- "topic": indicato con z, ogni parola viene generata da uno dei K topic del modello.

Queste entità, che corrispondono in termini statistici a variabili osservabili, come ad esempio le parole, o a variabili nascoste, come i topic, sono in relazione tra loro secondo funzioni di distribuzione di probabilità. Appare utile quindi un breve richiamo ad alcuni concetti di statistica che verranno spesso richiamati nei paragrafi che seguono.

3.1.3 Distribuzione multinomiale

In teoria delle probabilità la distribuzione multinomiale è una distribuzione di probabilità discreta che generalizza la distribuzione binomiale in più variabili. In altri termini, laddove la distribuzione binomiale descrive il numero di successi in un processo di Bernoulli, ad esempio il lancio di una moneta, per il quale ogni singola prova può fornire due soli risultati, la distribuzione multinomiale descrive il caso in cui ogni prova possa fornire diversi risultati con diverse probabilità.
Dati k possibili risultati della prova, ciascuno dei quali ha probabilità p_1, p_2, $\dots p_k$ di verificarsi ($\sum_{i=1}^{k} p_i = 1$), si eseguono n prove indipendenti. Se le variabili casuali X_i indicano il numero di volte in cui il risultato i è stato ottenuto su n prove, il vettore $\mathbf{X} = (X_1, \dots, X_k)$ segue una distribuzione multinomiale a parametri n e $\mathbf{p} = (p_1, \dots, p_k)$. La funzione di probabilità risulta essere pertanto:

$$\begin{aligned} f(x_1, \dots, x_k; n, p_1, \dots, p_k) &= \Pr(X_1 = x_1 \text{ e } \dots \text{ e } X_k = x_k) \\ &= \begin{cases} \dfrac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k}, & \text{se } \sum_{i=1}^{k} x_i = n \\ \\ 0 & \text{altrimenti} \end{cases} \end{aligned} \tag{3.1}$$

3.2 pLSA(pLSI)

Uno dei primi modelli generativi utilizzato per la classificazione di immagini è **pLSA - probabilistic Latent Semantic Analysis**, noto anche come *probabilistic LSI (pLSI).*
Secondo questo modello, considerando le osservazioni come co-occorrenze di

parole e documenti, la probabilità di ciascuna co-occorrenza viene modellata da una mistura di distribuzioni multinomiali *condizionatamente indipendenti*[2]. Ciascuna parola del documento è quindi un campione di un modello a mistura le cui componenti sono delle variabili casuali multinomiali corrispondenti ai topic. Pertanto, dato un topic z_i (non osservabile), un documento d ed una parola w_n sono in relazione secondo la legge:

$$p(d, w_n) = p(d) \sum_{i=1}^{k} p(w_n|z_i)p(z_i|d) \qquad t.c. \quad \sum_{i=1}^{k} p(z_i|d) = 1$$

La formula esprime la co-occorrenza della parola w_n e del documento d come prodotto tra la probabilità di estrarre il documento d dal corpus, $p(d)$, e la somma delle probabilità che ciascuno dei topic z_i generi w_n, $p(w_n|z_i)$, condizionata alla presenza del topic z_i nel documento d, ovvero $p(z_i|d)$. Questa ultima quantità, $p(z_i|d)$, può essere vista come il peso del topic i-esimo nel documento d.

In un modello a mistura di unigrammi ogni documento è generato a partire da argomenti estratti da un singolo topic. Il modello pLSI estende questa assunzione prevedendo la possibilità che un documento possa contenere più topic, distribuiti secondo una mistura di probabilità regolata dal termine $p(z|d)$[5]. Vale la pena far notare che con d si indica un documento appartenente al training set; l'indice d diventa a sua volta una variabile multinomiale con tanti valori quanti sono i documenti del training set: il sistema apprende le misture di topic $p(z|d)$ unicamente sugli elementi di questo set.

Per questo motivo, pLSI non è un modello generativo in senso stretto: le distribuzioni di topic $P(z|d)$ sono specifiche dei documenti di training da cui sono state empiricamente derivate e ciò non consente, pertanto, di assegnare in maniera naturale dei valori di probabilità ad un documento non visto. Un altro difetto di pLSI è che il numero di parametri da stimare cresce linearmente con il numero di documenti del training set. I parametri per un pLSI con k-topic sono dati da k distribuzioni multinomiali di dimensioni V (numero dei termini del vocabolario) e da M misture (numero dei documenti)

[2]Due eventi $A, B \in F$ sono condizionatamente indipendenti se dato $D \in F : (D) \neq 0$ si ha $P(A \bigcap B|D) = P(A|D)P(B|D)$

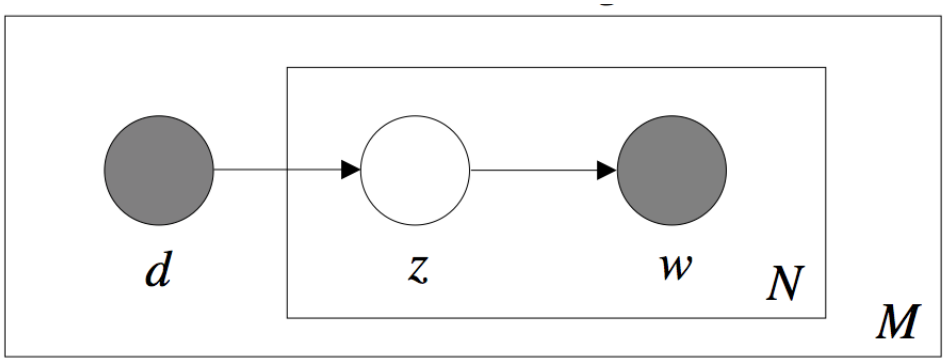


Figura 3.2. Modello grafico di pLSI: d rappresenta uno degli M documenti del corpus, z un topic e w una parola osservata nel testo. $P(z|d)$ indica la distribuzione dei topic per il documento corrente, $P(w|z_i)$ la distribuzione delle parole rispetto al i-esimo topic. d e w sono entità osservabili, i topic z_i sono variabili latenti.

su K topic nascosti. Si ottiene così un numero di parametri pari a $kV + kM$, ovvero lineare nel numero dei documenti di training, che suggerisce inoltre una tendenza del metodo ad incorrere in *overfitting*[3] sul set di dati.

3.3 LDA

Prima di passare alla formulazione estesa di LDA è utile fare un breve richiamo al concetto di *scambiabilità* che accomuna questo modello generativo ai modelli classici bag-of-words.

[3]In statistica, si parla di overfitting, ovvero "eccessivo adattamento", quando un modello statistico si adatta ai dati osservati, il campione, usando un numero eccessivo di parametri. In alcune circostanze, soprattutto nei casi in cui l'apprendimento è stato effettuato troppo a lungo o con uno scarso numero di esempi di allenamento, il modello potrebbe adattarsi a caratteristiche che sono specifiche solo del training set, ma che non hanno riscontro nel resto dei casi; per questo motivo, in presenza di overfitting, le prestazioni sui dati di training aumentano, mentre le prestazioni sui dati non visionati diventano peggiori.

3.3.1 Scambiabilità

Ogni documento del corpus può essere considerato come una sorta di"bag of words", ovvero un insieme disordinato di parole di un vocabolario: come per le applicazioni di IR, infatti, anche per LDA l'ordine in cui compaiono le parole nel documento non ha importanza. Implicitamente si assume che anche l'ordine con il quale i topic ed i documenti compaiono rispettivamente nei documenti stessi e nel corpus sia irrilevante; questi concetti sono tutti formalizzati nella definizione di "scambiabilità":

Definizione 3.3.1 *Un insieme finito di variabili casuali* $\{z_1, z_2, \ldots, z_N\}$ *viene detto* ***scambiabile*** *se la distribuzione congiunta è invariante a permutazione. Se* π *è una permutazione degli interi da 1 a N:*

$$p(z_1, z_2, \ldots, z_N) = p(z_{\pi(1)}, z_{\pi(2)}, \ldots, z_{\pi(N)})$$

Una sequenza infinita di variabili casuali è **infinitamente scambiabile** se ogni sottosequenza finita è scambiabile.
Il teorema di de Finetti (1990) stabilisce che un insieme di variabili casuali interscambiabili possono essere rappresentate con un modello a mistura - in generale anche infinita.

Teorema 3.3.1 (Teorema della rappresentazione di de Finetti) *Delle variabili casuali infinitamente scambiabili possono essere considerate indipendenti ed identicamente distribuite, condizionatamente ad un parametro casuale* θ *ottenuto da una qualche distribuzione.*

$$p(z_1, z_2, \ldots, z_N) = \int p(\theta) \left(\prod_{n=1}^{N} p(z_n|\theta) \right) d\theta \tag{3.2}$$

Il modello LDA assume che le parole siano generate dai topic secondo delle probabilità condizionate e che i topic siano infinitamente scambiabili in ciascun documento; il termine θ presente nell'equazione rappresenta il parametro casuale di una distribuzione multinomiale sui topic, modellata in LDA con la distribuzione di Dirichlet.

3.3.2 La distribuzione di Dirichlet

La distribuzione di Dirichlet appartiene alla famiglia delle funzioni esponenziali definite sul simplesso. In geometria per "simplesso" si intende una generalizzazione del concetto di triangolo o tetraedro di dimensione arbitraria. In particolare, un n-simplesso viene definito come politopo[4] n-dimensionale ottenuto come inviluppo complesso di $n+1$ vertici.
Formalmente un n-simplesso standard è un subset di $\mathbb{R}^{n+1}$ dato da:

$$\triangle^n = \{(t_0, \ldots, t_n) \in \mathbb{R}^{n+1} \mid \sum_{i=0}^{n} t_i = 1 \ e \ t_i \geqslant 0 \ \forall i\}$$

La funzione di densità di probabilità di Dirichlet è data dalla seguente formula:

$$p(\theta|\overrightarrow{\alpha}) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_k)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \theta_1^{\alpha_1 - 1} \cdots \theta_k^{\alpha_k - 1} \tag{3.3}$$

Si noti che nella formula in esame θ è una variabile casuale di Dirichlet K-dimensionale che assume valori nel simplesso di dimensione $k-1$; il parametro a valori positivi α ha anch'esso dimensione k; Γ può essere vista come un'estensione a valori reali della funzione fattoriale. Il rapporto $\frac{\Gamma(\sum_{i=1}^{k} \alpha_k)}{\prod_{i=1}^{k} \Gamma(\alpha_i)}$ assicura il rispetto del vincolo per cui l'integrale della funzione deve essere uguale ad 1.
Nella teoria di Bayes, la probabilità a posteriori di un evento casuale, o semplicemente *posterior*, è la probabilità assegnata all'evento dopo aver considerato l'evidenza ottenuta dall'osservazione di un esperimento.
Data una probabilità a priori $p(\theta)$, un'osservazione X ed un valore di likelihood $p(X|\theta)$[5], la relazione tra la probabilità a posteriori e quella a priori è esprimibile come:

$$p(\theta|X) \propto p(\theta) \cdot p(X|\theta)$$

[4]Definito nello spazio euclideo a più di 3 dimensioni è l'analogo di un poligono nel piano e di un poliedro nello spazio.

[5]La *likelihood* è una funzione dei parametri di un modello statistico, definita come segue: la likelihood di un set di parametri, data l'osservazione dei risultati di un esperimento, è uguale alla probabilità di ottenere i risultati osservati dati i parametri del modello.

La distribuzione di Dirichlet gode della proprietà di essere la coniugata[6]. della distribuzione multinomiale, ovvero data un'osservazione multinomiale, la distribuzione a posteriori di θ è una Dirichlet.
Il parametro α influenza la forma media (*mean shape*) di θ ed il grado in cui essa risulta essere sparsa, ovvero il numero di elementi nello spazio di distribuzione che hanno un valore alto di probabilità.
La figura 3.3 mostra alcuni esempi della distribuzione in esame.

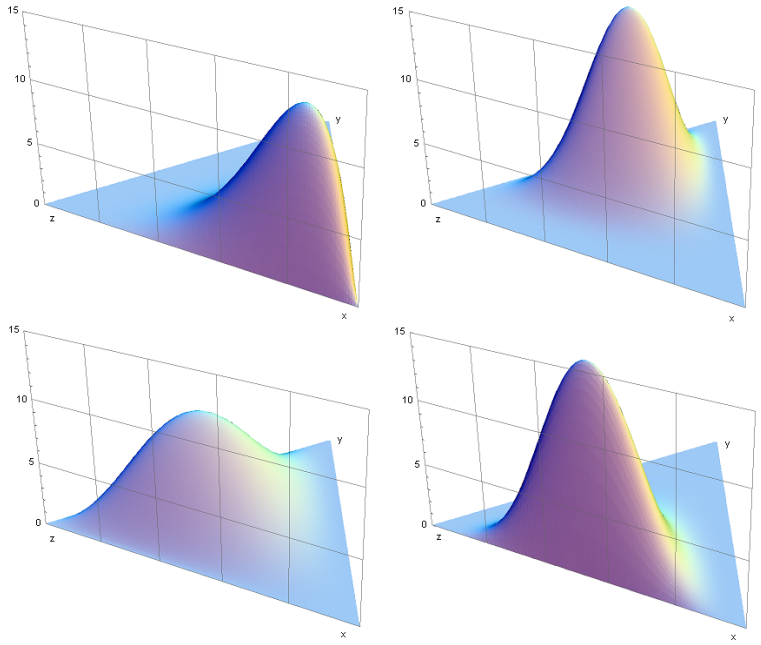


Figura 3.3. Alcuni esempi della distribuzione di Dirichlet per k=3 per differenti combinazioni dei parametri α. In senso orario partendo dall'angolo sinistro in alto: α=(6, 2, 2), (3, 7, 5), (6, 2, 6), (2, 3, 4).

Nel caso generico di una distribuzione con parametri $\theta \sim Dir(a, b, c)$, immaginando il triangolo avente come vertici gli elementi su cui la distribuzione è definita, la probabilità dei punti su un lato del triangolo avrà contributi unicamente dai due elementi agli estremi del lato (diventa difatti una distribuzione binaria, con il vertice non interessato avente probabilità uguale a

[6]Se la probabilità a priori è nella stessa famiglia della probabilità posteriori, allora le due distribuzioni si dicono coniugate.

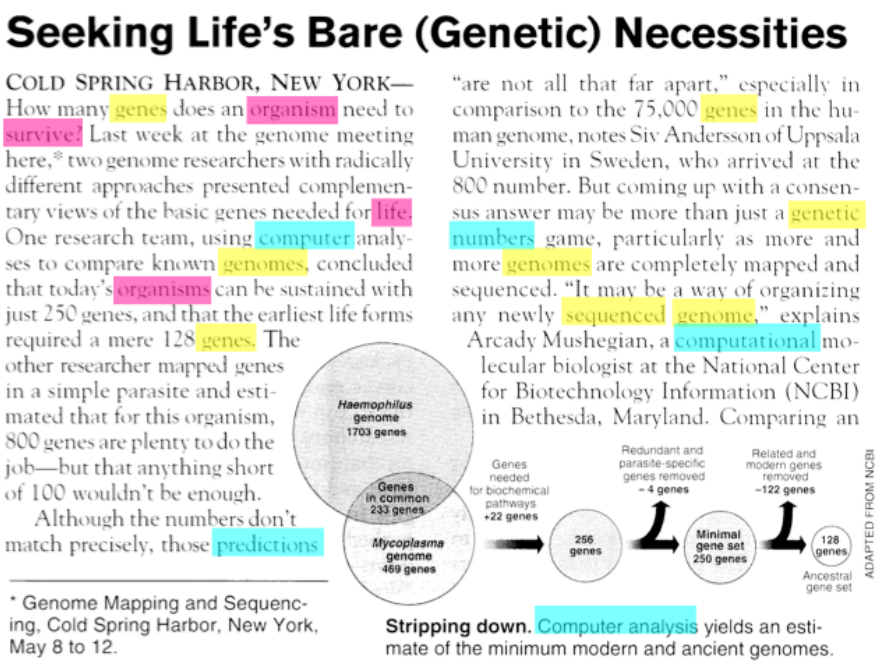
0), mentre la probabilità dei punti interni avrà una componente su tutti e 3 gli elementi. Nel caso particolare $\theta \sim Dir(1,1,1)$ ($\alpha_1 = \alpha_2 = \alpha_3 = 1$) sono unicamente definiti i punti all'interno del triangolo, con distribuzione di probabilità uniforme. Nel caso di $\theta \sim Dir(5,5,5)$ la distribuzione presenta un picco nel centro del simplesso. In generale il valore atteso dell'i-esimo componente di θ dato α si esprime come:

$$E[\theta_i|\alpha] = \frac{\alpha_i}{\sum_i \alpha_i}$$

e quindi in entrambi i casi visti il valore atteso per ciascun elemento della distribuzione è uguale ad 1/3. Differenziando i valori che compongono il vettore α il picco della distribuzione si sposterà dal centro del simplesso verso uno dei vertici; il valore massimo e la pendenza del picco cambiano al variare di $\sum_i \alpha_i$, che assume il significato di "dispersione statistica": un basso valore di $\sum_i \alpha_i$ denota un'elevata dispersione.
Le considerazioni appena fatte valgono nel caso in cui ciascun α_i sia $\geqslant 1$; se invece tutti i valori α_i sono inferiori ad 1, come nel caso della *Exchangeable Dirichlet* ($\theta \sim Dir(\alpha, \alpha, \alpha, \ldots, \alpha)$), gli elementi della distribuzione risultano sparsi e soltanto alcuni hanno una probabilità $\geqslant 0$: il numero di elementi aventi "massa" positiva diminuisce con il tendere di α a 0.

3.3.3 Formulazione di LDA

L'idea di base su cui si fonda LDA è che ogni documento contiene uno o più topic [5]. A partire da un modello di tipo generativo, ogni documento è considerato come una mistura casuale di topic, sui quali è definita una distribuzione di probabilità. I topic sono indipendenti dal documento e per ciascuno di essi esiste una distribuzione di probabilità sulle parole di un vocabolario: ogni parola può essere generata dai topic per mezzo di distribuzioni condizionate fissate; ciascun argomento (o topic) contiene diverse parole, ognuna con un proprio valore di probabilità. Per semplificare possiamo pensare al documento come ad un insieme disordinato di termini; i topic del documento vengono individuati osservando le occorrenze dei termini al suo interno e confrontandole con le distribuzioni dei termini per ciascun topic. La figura



Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

Figura 3.4. Topic contenuti in un documento

3.4 mostra come esempio un testo in cui con uno stesso colore (ad esempio il colore azzurro) sono individuati dei termini appartententi ad uno stesso topic ("computer", "numbers", "computational").
LDA ha l'obiettivo di inferire la struttura dei topic, determinare l'insieme dei topic associati ad un documento formalizzando una distribuzione di probabilità e, per ogni parola, stabilire il topic da cui è stata estratta: il risultato è un insieme di probabilità condizionate sulla variabili del sistema data l'osservazione dei termini che compaiono nel documento.

La figura 3.5 mostra LDA nella sua rappresentazione mediante modello grafico.
Sorvolando per il momento sul significato di α, che è un vettore k-dimensionale ed η, che è uno scalare, analizziamo nel dettaglio le altre componenti del modello, muovendoci da quelle più generali alle più specifiche, supponendo, come suggerisce la notazione del modello grafico, di avere un corpus di D documenti contenenti parole prese da un vocabolario di dimensione V e di

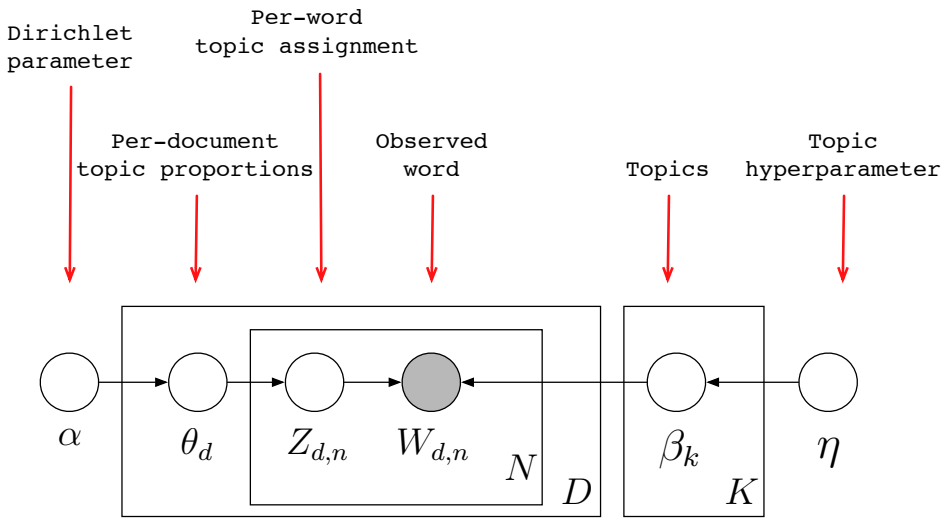


Figura 3.5. LDA graphical model

voler individuare le distribuzioni di K topic:

β_k per ogni topic esiste un termine β, ovvero un parametro della distribuzione Dirichlet a priori per-corpus, avente dominio definito sul simplesso di dimensione V: in particolare ciascuna β_k associa al topic k una distribuzione di Dirichlet sui termini del vocabolario;

θ_d indica le proporzioni con cui ciascuno dei K topic è presente nel documento $d \in D$, dove D rappresenta il corpus; per ciascun documento viene definito un vettore θ_d di dimensione K;

$Z_{d,n}$ assegnazione di un topic per la parola n nel documento d: dipende da θ_d poichè viene estratta dalla sua distribuzione di probabilità ed è un numero compreso tra 1 e K;

$W_{d,n}$ l'unica variabile casuale osservabile del modello: dipende da $Z_{d,n}$ e da β ed indica che la parola n è presente nel documento d.

Supponendo di conoscere le variabili nascoste, la probabilità di osservare una parola in un documento può essere individuata considerando l'elemento

$W_{d,n}$-esimo del termine $Z_{d,n}$-esimo dai $\beta_{1...K}$.

$$p(W_{d,n}|Z_{d,n}, \beta_{1...K}) = \beta_{Z_{d,n},W_{d,n}}$$

Si può rappresentare la relazione appena descritta sotto forma di matrice in cui le parole del dizionario V sono disposte lungo le righe e l'insieme dei topic è disposto lungo le colonne. La somma degli elemento lungo ciascuna colonna è uguale ad 1. Considerando il termine $W_{d,n}$ relativo alla parola ed il termine $Z_{d,n}$ riguardante il topic da cui è estratta la parola, l'elemento della matrice che rappresenta la probabilità di osservare quella parola nel documento viene individuata dal valore della cella $\beta_{Z_{d,n},W_{d,n}}$.
Bisogna precisare che nel caso ideale di variabili note i termini $Z_{d,n}$, piuttosto che essere modellate da una distribuzione di probabilità, assumerebbero un singolo valore compreso tra $1, .., K$; tuttavia le variabili in questione sono in genere non osservabili e sono pertanto rappresentate da distribuzioni; il termine θ_d è in ogni caso una distribuzione di probabilità. Di seguito viene riportata in formule la probabilità congiunta tra le variabili del modello:

$$\left(\prod_{k=1}^{K} p(\beta_k|\eta)\right) \cdot \left(\prod_{d=1}^{D} p(\theta_d|\alpha) \left(\prod_{n=1}^{N} p(Z_{d,n}|\theta_d) p(W_{d,n}|Z_{d,n}, \beta_{1...K})\right)\right) \quad (3.4)$$

La definizione delle variabili, nascoste ed osservate, in termini di distribuzione di probabilità è la seguente. Per quanto riguardi i topic:

- $p(\beta_k|\eta)$ rappresentano distribuzioni di Dirichlet V-dimensionali, sono indipendenti in quanto i parametri β_k sono unicamente dipendenti da η che è un parametro della distribuzione;

mentre per quanto riguarda i documenti:

- $p(\theta_d|\alpha)$ sono distribuzioni di Dirichlet K-dimensionali, dipendenti dal parametro α, si tratta anche in questo caso di distribuzioni definite a loro volta su altre distribuzioni (θ_d in questo caso);

infine all'interno di ciascun documento:

- $p(Z_{d,n}|\theta_d)$ rappresenta il valore di probabilità assegnato al $Z_{d,n}$-esimo indice all'interno di θ_d;

- $p(W_{d,n}|Z_{d,n}, \beta_{1...K})$ definita in precedenza.

Sappiamo che θ_d è una distribuzione di probabilità che modella le proporzioni in cui i K topic sono presenti nel documento d: da questa considerazione si può quindi dedurre che $p(Z_{d,n}|\theta_d)$ non è altro che il valore di probabilità per il topic $Z_{d,n}$-esimo. Possiamo scrivere quindi le seguenti relazione:

$$p(Z_{d,n}|\theta_d) = \theta_{d,Z_{d,n}}$$

La matrice dei topic β a dimensione $V \times K$ può essere rappresentata come una concatenazione di colonne lunghe V, ottenute da una distribuzione Dirichlet scambiabile a parametro η. In realtà questa modellazione di β è stata introdotta nella versione *smoothed* del modello LDA a cui il modello grafico si riferisce, in cui si permette l'assegnazione di probabilità non nulle a tutti gli elementi del vocabolario, anche quelli che eventualmente non comparirebbero nel training set di un esperimento. Per ovviare ai problemi di dispersione che si verificano in presenza di vocabolari di grandi dimensioni, quindi, si evita di assegnare probabilità nulla a nuovi documenti che contengono parole non usate durante l'addestramento.
L'equazione 3.4 mostra la probabilità congiunta di tutte le entità del modello supponendo note sia le variabili osservabili che quelle non osservabili: in realtà gli unici dati "misurabili" sul dataset di un esperimento sono le co-occorrenze documento-parola, da cui in una seconda fase è possibile inferire un modello LDA.
In particolare a partire da una collezione di documenti si vuole dunque ricavare:

- l'assegnazione dei topic per le parole di ciascun documento, ovvero i valori dei vari $Z_{d,n}$;
- le proporzioni dei topic per ogni documento, ovvero i vettori θ_d: il vettore K-dimensionale contenente le proporzioni dei topic del documento d può essere usato come descrittore del documento stesso;
- le distribuzioni degli argomenti su tutto il corpus (β).

I descrittori dei documenti, ovvero i valori attesi delle probabilità a posteriori di avere i K topic all'interno di ciascun documento d, possono poi essere usati per diversi scopi: operazioni di information retrieval, calcolo della similarità tra documenti, ecc. Per ottenere le quantità specificate è necessario quindi analizzare i documenti del corpus e per ciascuno di essi ricavare la corrispondente distribuzione θ_d utilizzando l'inferenza a posteriori.

$$E[\theta_d | W_{d,n}, \beta_{1...K}]$$

Al termine dell'analisi di tutti i documenti si otterrà un modello probabilistico della distribuzione degli argomenti nel corpus con una tecnica *non supervisionata*, in quanto non esiste nessuna conoscenza ed etichettatura dei topic durante la fase di addestramento. Uno dei punti di forza dell'analisi dei documenti per mezzo di un modello come LDA è che, nel momento conclusivo in cui si è creato il modello, le parole che appartengono ad un argomento sono in genere connesse tra di loro da un punto di vista "semantico". Questo comportamento è dovuto ad alcuni motivi:

- le probabilità delle parole vengono massimizzate distribuendo le parole tra gli argomenti; questo significa che non può accadere che si assegni a tutte le parole in ciascun topic un valore di probabilità proporzionale al numero di occorrenze che si misura su tutto il corpus in quanto l'evidenza farà sì che le parole si distribuiscano "a blocchi" tra i diversi argomenti.

- per un modello a mistura l'osservazione precedente è sufficiente per aspettarsi di trovare cluster di parole che si presentano sempre insieme (co-occorrenza);

- in LDA l'uso della distribuzione di Dirichlet sulle proporzioni di topic incoraggia la dispersione: un documento sarà penalizzato in caso contenga molti argomenti diversi; più i valori di α si avvicinano a 0, maggiore è la probabilità che sia presente un solo argomento per documento;

- in un certo senso ciò può essere visto come un rilassamento della definizione di "co-occorrenza" per modelli a mistura.

- Questa flessibilità porta ad avere insiemi di parole che molto probabilmente compariranno sempre insieme.

Quindi, in definitiva, considerando i documenti testuali o visuali come frutto di un processo generativo che porta a scegliere i termini in base a dei topic di interesse per il documento stesso, LDA stabilisce quali insiemi di parole hanno più probabilità di co-occorrenza e così facendo individua gli argomenti che hanno generato il documento.
Il modello di LDA appena proposto è la base di partenza su cui è possibile sviluppare in maniera modulare dei modelli più complessi; inoltre è necessario precisare che la distribuzione che genera i dati (β_k) può essere di qualunque genere, non necessariamente una Dirichlet.
Nei paragrafi che seguono viene proposta una breve trattazione dei metodi matematici per la stima delle grandezze del modello, ovvero le probabilità a posteriori che consentono di utilizzare il modello generativo come criterio di descrizione e classificazione di collezioni di dati.

3.3.4 Calcolo della probabilità a posteriori

Assumendo che gli argomenti $\beta_{1:K}$ siano fissati, la distribuzione a posteriori per ogni documento per le variabili non osservabili θ e Z_n è data dalla formula:

$$p(\theta, Z_{1:N}|W_{1:N}) = \frac{p(\theta|\alpha)\prod_{n=1}^{N} p(Z_n|\theta)p(W_n|Z_n, \beta_{1:K})}{\int_\theta p(\theta|\alpha)\prod_{n=1}^{N}\sum_{z=1}^{K} p(Z_n|\theta)p(W_n|Z_n, \beta_{1:K})} \tag{3.5}$$

Questa distribuzione, formalmente una "funzione ipergeometrica multipla", risulta intrattabile per via dell'impossibilità di disaccoppiare i termini θ e β nei successivi calcoli che, per brevità, non vengono riportati. In ogni caso si può vedere la formula 3.5 come la somma di N^K termini integrali di Dirichlet (trattabili). Esistono varie approssimazioni per il suo calcolo: metodi variazionali, usati ad esempio in [11] ed in [20]; campionamento di Gibbs, usato ad esempio in [24], [3], [38] ed in [4]; particle filtering.

Inferenza variazionale

Considerando in prima approssimazione β una matrice a valori costanti (una multinomiale), l'obiettivo principale del problema di inferenza per LDA è calcolare la distribuzione a posteriori delle variabili nascoste date le parole osservate per ogni documento:

$$p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(\mathbf{w}|\alpha, \beta)} \tag{3.6}$$

Questo significa che a partire dalle parole osservate nel documento, noti α e β, si vuole calcolare la distribuzione di probabilità degli argomenti nel documento θ, ricordando che per ogni documento d esiste un vettore K-dimensionale θ_d e gli argomenti associati ad ogni parola $\mathbf{z}$, inteso come vettore dei $Z_{d,n}$. Dato che questa distribuzione a posteriori non è computabile, si può cercare una limitazione inferiore della log likelihood $p(\mathbf{w}|\alpha, \beta)$. Il metodo di inferenza variazionale consiste nel considerare una famiglia di limitazioni inferiori indicizzata su un insieme di parametri e stabilire i valori dei parametri che minimizzano la distanza tra la funzione approssimata e quella reale.
Nel caso in esame vengono introdotti dei parametri variazionali liberi per modellare θ e $\mathbf{z}$. La distribuzione di probabilità associata al modello approssimato è della forma:

$$q(\theta, \mathbf{z}|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n)$$

I parametri γ e $\phi_1 \dots \phi_N$ sono rispettivamente di tipo Dirichlet e multinomiale. I valori ottimi dei parametri sono quelli che minimizzano la divergenza Kullback-Leibler tra la vera distribuzione a posteriori e quella approssimata:

$$(\gamma^*, \phi^*) = argmin_{(\gamma,\phi)} D(q(\theta, \mathbf{z}|\gamma, \phi)||p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta)) \tag{3.7}$$

Il processo di minimizzazione viene eseguito iterativamente secondo il principio del *punto fisso*; le equazioni di aggiornamento dei parametri calcolate risolvendo il problema appena descritto sono le seguenti:

$$\begin{aligned} \phi_{ni} &\propto \beta_{iw_n} exp\left\{E_q\left[log(\theta_i)|\gamma\right]\right\} \\ \gamma_i &= \alpha_i + \sum_{n=1}^{N} \phi_{ni} \end{aligned} \tag{3.8}$$

Nello specifico si ha che:

- la variabile multinomiale ϕ viene aggiornata in base al teorema di Bayes: $p(z_n|w_n) \propto p(w_n|z_n)p(z_n)$; il vaolre di $p(z_n)$ è approssimato dalla funzione esponenziale;
- la Dirichlet γ viene aggiornata in base ai valori attesi del numero di parole per ogni topic nel documento, $E[z_n|\phi_n]$; in particolare il parametro $i-esimo$ della distribuzione a posteriori, γ_i, viene approssimato come la somma dell'$i-esimo$ parametro della distribuzione Dirichlet a priori, α_i, più il valore atteso del numero di parole generate dal topic $i-esimo$.

Da notare che la distribuzione variazionale sui parametri γ e ϕ è ancora una distribuzione condizionata su $\mathbf{w}$, in quanto il processo di ottimizzazione viene condotto tenendo fisso questo termine.
Stabiliti i valori ottimi per i parametri γ e ϕ, associati ad un'approssimazione ottimale della distribuzione a posteriori, è necessario calcolare i valori dei parametri α e β. Essi devono essere tali da massimizzare la log likelihood dei dati:

$$l(\alpha, \beta) = \sum_{d=1}^{M} \log p(\mathbf{w}_d|\alpha, \beta) \tag{3.9}$$

Sostituendo a $p(\mathbf{w}|\alpha, \beta)$ l'approssimazione calcolata tramite inferenza variazionale, è possibile calcolare i valori di α e β che massimizzano la log likelihood.
Iterando i passi descritti fino alla convergenza dell'approssimazione della log likelihood si ottiene una procedura di EM (Expectation Maximization) variazionale per la stima del modello LDA:

- (E-step) per ciascun documento vengono calcolati i valori ottimali di (γ^*, ϕ^*) con il procedimento iterativo descritto in precedenza;
- (M-step) calcolo dei parametri α e β che massimizzano il limite inferiore della funzione di log likelihood.

Questi due passi sono ripetuti fino alla convergenza della funzione trattata nel M-step.

Campionamento di Gibbs

Il campionamento di Gibbs è un altro metodo usato per l'approssimazione delle distribuzioni a posteriori del modello LDA; se ne riporta qui di seguito una breve descrizione che, pur non essendo esaustiva, ha il semplice scopo di illustrare i principi di funzionamento. Questo metodo si basa sul principio secondo cui è più semplice campionare su distribuzioni di probabilità condizionata piuttosto che marginalizzare integrando su funzioni di probabilità congiunta.In particolare, a partire da una catena di Markov stazionaria[7] MC, si approssima la posterior di interesse con i campioni raccolti sul MC; lo spazio definito dalla MC rappresenta le possibili configurazioni delle variabili nascoste. L'idea di base è che lo stato futuro di una data variabile si ottiene campionando la stessa sulla propria distribuzione condizionata ai valori correnti di tutte le altre variabili del sistema.
Il procedimento si articola essenzialmente in due fasi:

1. dato $Z_{1:N}$ lo stato attuale delle variabili nascoste, Z_{-i} lo stato degli altri elementi e $n(Z_{1:N})$ un vettore contatore del numero di occorrenze dei topic, si definisce:

$$p(\theta|Z_{1:N}, W_{1:N}) = Dir(\alpha + n(Z_{1:N}))$$

$$p(Z_i|Z_{-i}, W_{1:N}) = Mult(\pi(Z_{-i}, W_i))$$

$$dove\ \pi(Z_{-i}, W_i)) \propto (\alpha + n(Z_{1:N}))p(W_i|\beta_{1:K})$$

2. calcolati i nuovi valori di θ si aggiorna il valore di ciascun Z_i in base ai Z_{-i} (collapsed Gibbs sampling):

$$p(Z_i|Z_{-1}, W_{1:N}) \propto p(W_i|\beta_{1:K}) \prod_{k=1}^{K} (\Gamma(n_k(Z_{-i}))$$

[7]Per una catena di Markov stazionaria si ha che la probabilità di passare da uno stato a ad uno stato b rimane immutata nel tempo.

dove $n_k(Z_{-i})$ indica il numero di volte in cui il topic k è presente nell'assegnazione di topic Z_{-i}.

3.3.5 Smoothing

Come anticipato durante la descrizione del modello grafico di LDA, l'introduzione dello *smoothing* evita di assegnare probabilità nulle a termini che, pur appartenendo ad un vocabolario precomputato, non vengono mai visti nel training set di un esperimento. In effetti un nuovo documento potrebbe verosimilmente contenere parole non presenti nel corpus di addestramento, le quali riceverebbero probabilità nulla nel processo di massimizzazione della likelihood per i parametri delle distribuzioni multinomiali e porterebbero ad avere documenti con probabilità nulle. La soluzione a questo inconveniente consiste nell'assegnare probabilità non nulle a tutti i termini, a prescindere dalla loro effettiva presenza nel training set.
Supponendo per β è una rappresentazione mediante una matrice a valori casuali $V \times K$, con una colonna per ogni componente della mistura, si deriva ciascuna colonna da una distribuzione Dirichlet scambiabile a parametro η, ottenendo in base alla definizione data nel paragrafo 3.3.2 una Dirichlet V-dimensionale con $\alpha_i = \eta \, \forall i$.
Anche in questo caso la procedura di inferenza può seguire un approccio variazionale, definendo in aggiunta una distribuzione per le variabili β, θ e $\mathbf{z}$:

$$q(\beta_{1:k}, \mathbf{z}_{1:M}, \theta_{1:M} | \lambda, \phi, \gamma) = \prod_{i=1}^{k} Dir(\beta_i | \lambda_i) \prod_{d=1}^{M} q_d(\theta_d, \mathbf{z}_d | \phi_d, \gamma_d) \tag{3.10}$$

Le equazioni di aggiornamento per γ e ϕ sono le stesse viste in precedenza, mentre l'equazione che aggiorna il nuovo parametro λ è la seguente:

$$\lambda_{ij} = \eta + \sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi^*_{dni} w^j_{dn} \tag{3.11}$$

In questo caso la procedura di EM variazionale massimizza la log likelihood rispetto ai parametri α e η.

Capitolo 4

Soluzione proposta

Nel capitolo 2 è stata fornita una descrizione teorica di una metodologia di base per l'analisi e la classificazione di documenti, ovvero la descrizione via bag of words fornita come input a metodi di classificazione Nearest Neighbor o SVM. Nel capitolo 3 si è visto l'approccio di modelli alternativi al bag of words, ovvero i modelli a topic latenti ed in dettaglio LDA. In particolare si è visto come un'analisi di tipo probabilistico porti ad una descrizione di livello semantico più alto, in quanto le parole visuali non vengono soltanto considerate come elemento statistico di cui misurare la frequenza in un vettore *tf-idf* ma, al contrario, diventano un elemento informativo dal quale inferire a livello probabilistico la struttura di uno o più topic "generatori".
La prima parte del lavoro di tesi è incentrata pertanto sul confronto delle due seguenti modalità:

- classificazione di immagini descritte attraverso bag of words;
- classificazione di immagini descritte per mezzo di modelli a topic latenti, in particolare mediante Latent Dirichlet Allocation.

Questo confronto ha lo scopo di misurare non soltanto le prestazioni assolute in termini di accuratezza di classificazione, ma anche i vantaggi in termini di riduzione dimensionale dello spazio necessario alla descrizione della collezione di dati.
In una seconda fase sono state introdotte altre due modalità di descrizione:

- Spatial Pyramid Matching (SPM) [37]: si tratta dell'applicazione del modello bag of words ad una rappresentazione delle immagini mediante una struttura piramidale, costruita suddividendo ricorsivamente ciascuna immagine in sottoregioni di dimensioni vie via più piccole. Di questa modalità si parlerà in maniera più diffusa ne paragrafo 4.2.3.

- GIST [30]: a differenza degli altri metodi in cui la raccolta delle feature di un'immagine viene condotta a livello locale, ovvero ricercando i punti salienti all'interno delle stesse, in questo caso viene usato un globale che ha lo scopo di catturare caratteristiche globali della scena rappresentata nelle immagini; questa modalità verrà affrontata nel paragrafo 4.2.4.

Ciascuna delle 4 modalità può essere usata singolarmente per effettuare classificazione di documenti; in ogni caso ciascuna di esse seguirà un pipeline di esecuzione schematizzata essenzialmente in questi 3 passi:

1. **raccolta delle feature**, che siano esse puntuali, ovvero tutte quelle derivanti dal SIFT, o globali, ad esempio quelle derivanti dal GIST;

2. **descrizione dei documenti**: ad ogni documento è associato un descrittore proveniente dall'applicazione del modello bow, come nel caso della baseline classica di bag of words o nel caso di SPM; oppure proveniente da un modello a topic latenti; o infine derivante da una descrizione globale come nel caso di GIST;

3. **classificazione**: i descrittori associati ai documenti costituiscono l'input per i metodi di classificazione che rimangono gli stessi per tutte le singole pipeline.

L'insieme di queste 4 modalità singole, dette anche *unimodal pipeline*, costituisce la base di partenza per il secondo obiettivo della tesi: misurare le prestazioni della classificazione per mezzo di tecniche di fusione di kernel, ovvero misurare l'accuratezza di un sistema che classifica mediante **Multiple Kernel Learning**.

4.1 Modalità di classificazione

Le pipeline unimodali introdotte nel paragrafo precedente si differenziano sostanzialmente per i metodi con cui viene condotta la descrizione dei documenti; le metodologie che riguardano la fase di apprendimento e classificazione, invece, sono identiche per tutte le modalità. Di seguito vengono descritti, sia per la classificazione basata su Nearest Neighbors che per quella basata su SVM, i criteri scelti per misurare le relazioni di similarità tra i documenti della collezione.

4.1.1 Metodi basati su Nearest Neighbors

Come già precisato nel paragrafo 2.6.1 i metodi basati su Nearest Neighbors assegnano un'etichetta ad un documento di test in base alla classe di appartenenza dei k elementi del training set che risultano essere più vicini. I valori del descrittore di ciascun elemento della collezione costituiscono le componenti del vettore associato ad esso nello spazio dei documenti; la distanza tra i documenti viene calcolata in base ad una qualche metrica.
Di seguito si riportano le varianti di k-NN utilizzate per gli esperimenti.

k-NN

Si tratta della metodologia di base, in cui ad un documento di test viene assegnata la classe più frequente nei k documenti di training set ad esso più vicini in base ad una metrica scelta. Supponendo ad esempio di avere un modello bag of words con un vocabolario di N termini, ciascun documento della collezione sarà descritto con un vettore a dimensione N. Dato un documento di test con il rispettivo descrittore, si determinano i k elementi del training a distanza minore nello spazio N-dimensionale dei documenti. Una naturale estensione di questa metrica consiste nel ricercare un solo elemento tra quelli più vicini, con $k = 1$, assegnando al documento di test la classe di appartenenza del documento trovato.

k-NN class

In questo metodo la ricerca degli elementi più vicini viene condotta con un approccio leggermente diverso: piuttosto che cercare i k elementi del training set che in termini assoluti risultano essere più vicini, vengono ricercati i k elementi più vicini per ciascuna classe e le distanze misurate con il documento di test vengono sommate per dare una stima della distanza *per-classe* e non più *per-documento*.
Questo metodo, suggerito in [29], ha il pregio di migliorare le prestazioni nei casi in cui il numero di esempi forniti per una data classe non è "proporzionale" alla complessità della classe stessa, ovvero alla variabilità nella rappresentazione degli oggetti che la compongono.

Metriche

Il calcolo delle distanze tra un documento di test x ed un documento del training set y viene effettuato secondo le seguenti metriche:

- distanza euclidea (L2):

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4.1}$$

- distanza χ^2, una metrica largamente usata in statistica, definita come:

$$\chi^2(\bar{x}, \bar{y}) = \frac{1}{2} \sum_i \frac{(x_i - y_i)^2}{x_i + y_i} \tag{4.2}$$

  dove

$$\bar{x} = [\bar{x_1} \dots \bar{x_n}] \quad \bar{x_i} = \frac{x_i}{\sum_{k=1}^{n} x_k}$$

Le funzioni appena viste sono impiegate non soltanto nei metodi k-NN, ma anche per le funzioni *kernel* calcolate nell'apprendimento mediante SVM.

4.1.2 Funzioni Kernel per l'apprendimento con SVM

Nel paragrafo 2.6.2 si è visto che, data una collezione di esempi appartenenti ciascuno ad una qualche categoria, è possibile per mezzo delle Support

Vector Machine individuare degli iperpiani di separazione nello spazio dei documenti. Il processo che porta al risultato ottimale si basa principalmente sulla minimizzazione del numero di esempi classificati erroneamente, o alternativamente sulla massimizzazione delle distanze tra i documenti di ciascuna classe e gli iperpiani di separazione (o meglio l'iperpiano di separazione, dato che la classificazione multiclasse non è altro che una classificazione 1-vs-all per ciascuna categoria).
L'applicazione di una SVM ad una collezione di dati porta ad avere una rappresentazione sotto forma di matrice delle relazioni di distanza, e quindi anche di similarità, tra tutte le coppie di documenti della collezione.
In particolare, dati due elementi x_i e x_j, il grado di similarità tra di essi viene calcolato per mezzo di una funzione *kernel* ; supponendo di avere T documenti la matrice $K = T \times T$ ottenuta applicando la funzione kernel a ciascuna coppia conterrà nella posizione (i, j) (con $i, j \leq t$) il valore di similarità associato alla coppia (x_i, x_j); gli elementi di K che si trovano sulla diagonale assumono tutti un valore pari ad 1.
Negli esperimenti condotti sono state utilizzate le seguenti funzioni *kernel*:

Kernel lineare

Data una coppia (x, y) di documenti descritti in uno spazio N-dimensionale, ad esempio per mezzo di una bag of words con vocabolario formato da N parole, si definisce la funzione $k(x, y)$ come il prodotto scalare tra i due vettori descrittori:

$$k(x, y) = x^T y \tag{4.3}$$

Kernel χ^2

Si tratta di una misura di similarità basata sulla distanza χ^2 descritta in precedenza, in particolare la funzione viene definita come:

$$k(x, y) = 1 - \chi^2(x, y) \tag{4.4}$$

Kernel exp(χ^2)

Questo kernel definisce una funzione esponenziale sulla distanza χ^2; la similarità tra la coppia (x, y) si definisce come

$$k(x, y) = \exp^{-\frac{1}{\gamma}\chi^2(x,y)} \tag{4.5}$$

dove γ è definito, in analogia ad [32], come il valore atteso delle distanze χ^2 sull'insieme delle coppie.

Kernel *intersection*

Il kernel *intersection* viene utilizzato spesso in computer vision e si dimostra particolarmente utile nelle operazioni di calcolo che coinvolgono rappresentazioni ad istogramma. La funzione kernel viene definita in questo caso come segue:

$$k(x, y) = \sum_{i=1}^{N} min\{x_i, y_i\} \tag{4.6}$$

4.2 Pipeline unimodali

In questa sezione viene fornita una descrizione logica delle pipeline unimodali utilizzate per gli esperimenti; ciascuna pipeline è relativa ad un particolare modello di descrizione delle immagini ed in alcuni casi, come ad esempio per il bag of words classico o per i modelli a topic latenti, è stata replicata più volte a seconda del tipo e della dimensione del descrittore.

4.2.1 Bag of Words

La figura 4.1 presenta un grafico riassuntivo dei blocchi che compongo il processo di classificazione con il modello bag of words.

La fase iniziale, indicata dal blocco con la dicitura "SIFT", consiste nella ricerca delle feature all'interno delle immagini: negli esperimenti condotti sono stati utilizzati, ad esempio, diversi tipi di descrittori SIFT, sia a scala di grigi che a colori, calcolati su uno o più valori di scala del filtro gaussiano. Ciascuna combinazione $<$ *tipo di descrittore*, *numero di scale* $>$ comporta

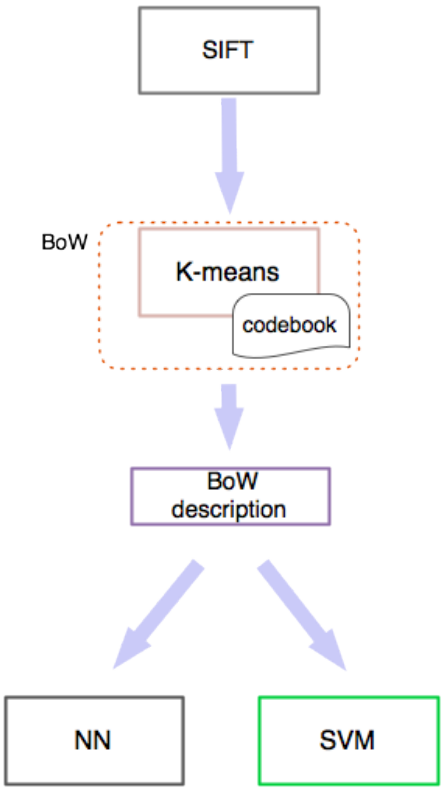


Figura 4.1. Pipeline di classificazione per il modello bag of words

l'esecuzione di una pipeline diversa, in quanto ciascuna delle scelte possibili porta ad una differente la modalità di rappresentazione delle informazioni individuate all'interno delle immagini.
Scelta una tipologia di descrizione, ad esempio SIFT monoscala a 128 dimensioni, ed un metodo di ricerca delle feature, ad esempio individuando i punti salienti su una griglia a spaziatura fissa, la fase di descrizione produce per ciascuna immagine un set di vettori di dimensione 128, uno per ciascun punto saliente.

La seconda fase del processo riguarda le operazioni di clustering e di quantizzazione. In particolare, fissata con un parametro N la dimensione del vocabolario o *codebook*, l'algoritmo k-means consente di individuare il set di

N cluster corrispondenti alle parole visuali del dizionario a partire da un insieme di vettori SIFT forniti come input. In particolare bisogna precisare che l'algoritmo non riceve in input i descrittori SIFT di tutte le immagini della collezione, ma soltanto un set di dimensione variabile individuato a sua volta tra i punti salienti delle immagini che formano il training set. La scelta di limitare il numero di descrittori SIFT con cui costruire il vocabolario, seguita per l'addestramento di tutte le pipeline basate su feature locali, viene dettata dalla necessità di trovare un giusto compromesso tra le dimensioni dei dati usati per il training e l'accuratezza ottenuta con il sistema addestrato su di essi.
Una volta calcolato il vocabolario, la descrizione finale di un'immagine di training o di test passa attraverso la fase di quantizzazione: data un'immagine I, tutti i vettori 128-dimensionali relativi ai punti salienti trovati in essa vengono etichettati con l'indice del cluster che si trova a distanza minore nello spazio delle feature a 128 dimensioni. La frequenza con cui ciascuna parola visuale è presente nell'immagine I viene riportata in un istogramma di dimensione N che rappresenta il descrittore definitivo dell'immagine. La dimensionalità dello spazio di descrizione dei documenti risulta quindi essere direttamente dipendente dalla grandezza del vocabolario: un valore maggiore di N consente di avere un numero più alto di parole per descrivere la collezione di dati; tuttavia più crescono le dimensioni dello spazio dei documenti e maggiore risulta il carico computazionale da sostenere per le operazioni di classificazione. Bisogna aggiungere inoltre che per qualsiasi configurazione di dataset e descrittori le prestazioni non aumentano indefinitamente con il crescere di N ma, al contrario, è generalmente possibile individuare un valore di saturazione oltre il quale le prestazioni non vengono più incrementate dall'aumentare delle dimensioni del vocabolario.

Avendo a questo punto ottenuto per ciascuna immagine una descrizione compatta di dimensione N, è possibile effettuare la classificazione degli elementi del test set sulla base dei dati raccolti durante la fase di apprendimento. Ciascuna pipeline viene eseguita su un valore fisso di N e permette di classificare le immagini del test set sia con metodi Nearest Neighbors che

con metodi SVM.
Nei metodi basati su Nearest Neighbor il descrittore N-dimensionale di un'immagine viene utilizzato per individuare i k elementi ad essa più vicini nello spazio dei documenti, mentre la conoscenza a priori delle classi di appartenenza per le immagini di training è sfruttata soltanto nel momento conclusivo in cui si deve assegnare la classe all'immagine di test. In questo senso i metodi NN sono stati usati in modo *semi-supervisionato*.
Nei metodi basati su SVM, al contrario, la conoscenza della classe di appartenenza per le immagini di training costituisce l'informazione attraverso cui si identificano gli iperpiani di separazione nello spazio dei documenti; tale conoscenza rende pertanto completamente *supervisionato* il processo di apprendimento. Date T_{train} immagini di training e T_{test} immagini di test, le relazioni di similarità vengono espresse da due matrici formate nel seguente modo:

- $K_{train} : T_{train} \times T_{train}$ per le coppie $< I_1, I_2 >$ tali che $I_1, I_2 \in Train\ Set$;
- $K_{test} : T_{test} \times T_{train}$ per le coppie $< I_1, I_2 >$ tali che $I_1 \in Test\ Set$ e $I_2 \in Train\ Set$;

Per ogni funzione kernel utilizzata è necessaria, prima dell'apprendimento vero e proprio, una fase di cross-validazione per la stima del valore ottimale del parametro di costo C.

Le considerazioni fatte sui metodi di classificazione rimangono immutate per le pipeline successive; anche se non esplicitamente dichiarato di volta in volta, si assume pertanto anche di seguito che i metodi di classificazione adottati rispettino la descrizione introdotta in questo paragrafo.

4.2.2 Latent Dirichlet Allocation

Nel capitolo 3 è stata fornita un'ampia descrizione formale dei metodi basati sull'analisi dei topic latenti; in particolare si è visto come dall'analisi delle parole visuali e testuali presenti in un documento sia possibile derivare, in termini probabilistici, la composizione degli argomenti che hanno generato il

contenuto del documento stesso.
L'obiettivo di questo paragrafo è illustrare come i modelli a topic latenti sono stati integrati nelle pipeline utilizzate per gli esperimenti. La figura 4.2 mostra lo schema a blocchi di una pipeline che fa uso di un modello LDA.

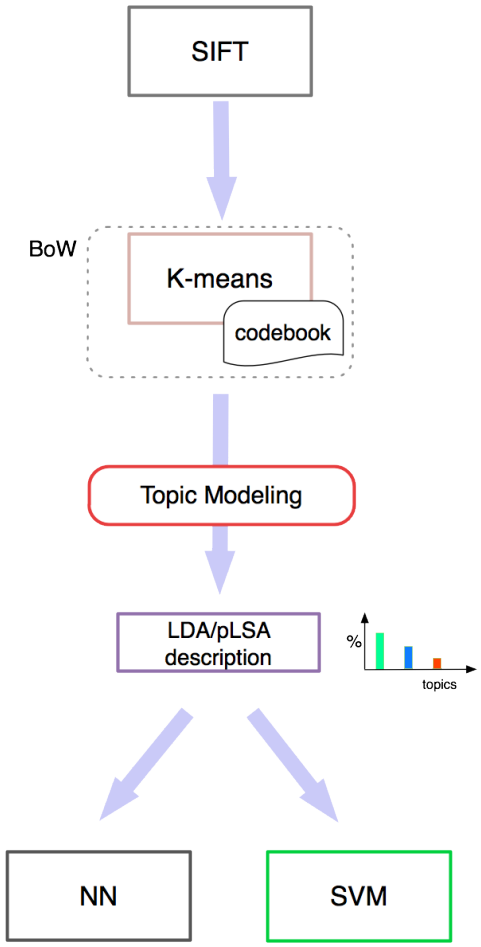


Figura 4.2. Pipeline di classificazione per la modalità con modelli a topic latenti

Facendo una veloce comparazione con lo schema visto nel paragrafo precedente, si nota come la pipeline usata per LDA condivida con quella relativa al BoW i seguenti passi:

- **raccolta delle feature**: le informazioni cercate all'interno dell'imma-

gine sono sempre quelle derivanti dalla descrizione di punti salienti; questo significa che a parità di parametri fissati per il modello LDA, è possibile replicare tante pipeline a topic latenti per quante diverse configurazioni $<$ *tipo di descrittore*, *numero di scale* $>$ vengono usate per la raccolta delle feature;

- **clustering-quantizzazione**: le informazioni utilizzate dai modelli a topic latenti come input per la costruzione del modello probabilistico riguardano le frequenze con cui le parole vengono individuate all'interno di ciascun documento; risulta necessario, pertanto, che si disponga di un vocabolario costruito su un set di descrittori mediante operazioni di clustering e di una fase di quantizzazione che codifichi ciascun descrittore SIFT con l'indice del cluster ad esso assegnato;

- **classificazione**: ai fini della classificazione l'output ottenuto da un modello a topic latenti viene trattato nello stesso modo in cui viene usato un descrittore di tipo bag of words, ovvero come un vettore di dimensione nota per ciascuna immagine da utilizzare per la stima delle distanze nello spazio dei documenti.

La differenza sostanziale tra questa pipeline e quella precedente consiste nel tipo di rappresentazione del documento che si intende utilizzare per stimare le relazioni di similarità e, di conseguenza, per classificare le immagini di test. Mentre il modello bag of words genera per ciascuna immagine un descrittore, di dimensione pari al codebook, che contiene la frequenza misurata con cui ciascuna parola compare all'interno di essa, il modello LDA fornisce per ciascun documento d un vettore θ_d di valori che esprimono la probabilità con cui i topic si distribuiscono al suo interno.
Questo processo presuppone quindi che per ciascuna immagine si conosca il set di osservazioni, ovvero il vettore descrittore N-dimensionale ottenuto tramite quantizzazione. A partire da questo viene generato in una fase successiva un vettore di dimensione Z pari al numero dei topic che verrà utilizzato dai metodi NN ed SVM per la sua collocazione all'interno dello spazio dei documenti Z-dimensionale.

I modelli a topic latenti forniscono una descrizione dei documenti che si colloca ad un livello semantico più alto rispetto a quella fornita dal modello bag of words. Infatti tali descrizioni non si limitano soltanto a registrare il numero di volte in cui gli elementi codificati, come i termini di un dizionario, sono visibili in un documento. Tentano invece di identificare i *concetti* che sono stati utilizzati per generare l'informazione: nel caso testuale, gli argomenti scelti dall'autore del documento; in quello visivo, gli oggetti che compongono la scena rappresentata.
Poichè i concetti rappresentabili in un documento prescindono dalla natura del documento stesso, ovvero testo o immagine, i modelli a topic latenti si prestano ad esperimenti di classificazione di collezioni di dati rappresentati in maniera eterogenea, ad esempio delle immagini a cui sono associati insiemi di tag [33].
Bisogna far notare, tuttavia, che i metodi di rappresentazione a modelli latenti sono di tipo *non-supervisionato*: questo significa che non è possibile individuare una corrispondenza $1:1$ tra un insieme di topic ed un insieme di concetti che si suppone essere presenti in un set di dati. Da ciò deriva anche il fatto che il numero di topic ottimale per un esperimento, come si vedrà nel capitolo dedicato ai risultati dei test, non è direttamente dipendente dal numero di classi della collezione: non è possibile instaurare una corrispondenza diretta topic - classe, ma le prestazioni ottenute mostrano che i topic si distribuiscono in maniera simile sugli elementi di una stessa classe.
In ogni caso si può notare come il numero di topic scelto per un esperimento determini la dimensionalità con cui si affronta il problema della descrizione della collezione e che, in molti casi, si riesca ad eguagliare e migliorare le prestazioni del modello bag of words con il vantaggio di una riduzione dimensionale pari al valore $\frac{N}{Z}$. Nell'immagine 4.3 viene fornito un esempio del concetto secondo cui, a partire da un set di immagini appartenenti a due classi diverse, alla fine dell'elaborazione con LDA ciascuna classe risulta avere una distribuzione di topic caratteristica.

Per completezza è necessario precisare che la quasi totalità degli esperimenti con questo tipo di pipeline sono stati condotti sul modello LDA

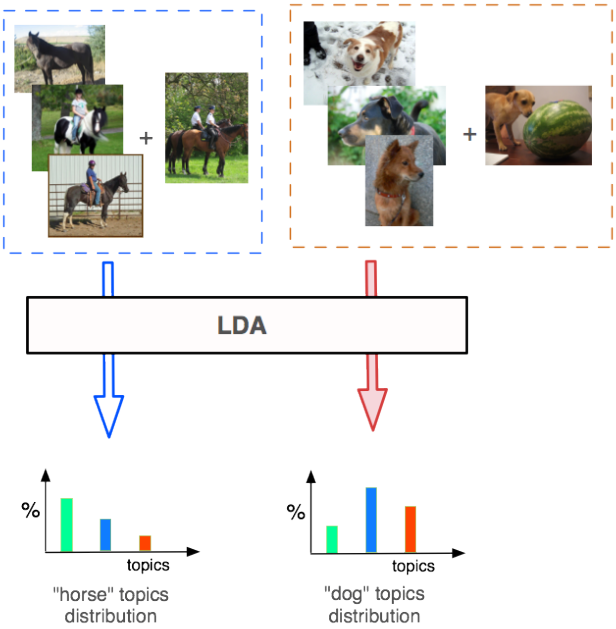


Figura 4.3. Esempio di distribuzione di $K = 3$ topic per le classi *horse* e *dog*

piuttosto che su quello pLSA, il quale ha mostrato delle prestazioni di classificazione leggermente inferiori al primo: questa preferenza trova anche una giustificazione teorica nel fatto che LDA, al contrario di pLSA, sia definito come modello completamente generativo in cui le distribuzioni dei topic per i documenti di test sono calcolate a partire da una distribuzione di probabilità definita su una parametro casuale piuttosto che essere vincolate alla matrice delle distribuzioni valutate empiricamente sui documenti di training. A tal proposito vale la pena far notare che la versione utilizzata per LDA è stata quella denominata *smoothed*, in cui si attribuiscono dei valori di probabilità diversi da 0 a termini non visti nella fase di training, permettendo così di avere una sorta di autoaggiornamento del modello anche durante l'analisi delle immagini di test.

4.2.3 Spatial Pyramid Matching

La terza pipeline utilizzata per gli esperimenti consiste in un'evoluzione del modello classico di bag of words. In questo nuovo modello la descrizione delle feature di ciascuna immagine avviene mediante quantizzazione su di un vocabolario visuale, arricchita dalla localizzazione spaziale dei punti salienti del documento. Prima di descrivere la collocazione del metodo di Spatial Pyramid Matching (SPM) all'interno di una pipeline di classificazione, vale la pena fare un breve richiamo dei concetti su cui si basa la sua formulazione.

Pyramid Match Kernel

ll metodo di Spatial Pyramid Matching presentato in [37] si basa sul principio del Pyramid Matching, formalizzato da Grauman e Darrell in [15]. Siano X, Y due set di vettori appartenenti ad uno spazio $d-$dimensionale, gli autori propongono il Pyramid Matching come criterio per trovare delle corrispondenze o matching tra tali set. Il principio di funzionamento è il seguente: lo spazio delle feature è suddiviso ricorsivamente in sottoregioni, o griglie, di grandezza via via minore; le corrispondenze trovate tra i due set vengono pesate in base al fattore di risoluzione della sottoregione nella quale avvengono tali matching. Ad ogni livello di risoluzione, infatti, due punti sono in corrispondenza se cadono all'interno della stessa cella; i match trovati a risoluzioni più elevate ricevono un peso maggiore rispetto a quelli individuati a risoluzioni meno fini.
Formalmente viene costruita una sequenza di griglie a risoluzione $0, \ldots, L$ in modo che la griglia a risoluzione l abbia 2^l celle per ciascuna dimensione; in totale si hanno $D = 2^{dl}$ celle, dove d indica il numero di dimensioni. Siano H_X^l e H_Y^l gli istogrammi di X e Y alla risoluzione l e $H_X^l(i)$ e $H_Y^l(i)$ il numero di punti di X e Y che cadono nella $i-$esima cella della griglia; il numero di match a livello l è dato dalla funzione di intersezione come segue :

$$I(H_X^l, H_Y^l) = \sum_{i=1}^{D} min(H_X^l(i), H_Y^l(i)) \tag{4.7}$$

Per brevità di seguito $I(H_X^l, H_Y^l)$ saranno indicati con I^l.
Si può notare come il numero di corrispondenze trovate a livello l includa

anche quelle trovate al livello di risoluzione più fine $l+1$, quindi il numero di nuovi match trovati a livello l può essere calcolato come $I^l - I^{l+1}$ per $l = 0, \ldots, L-1$. Il peso associato al livello l è pari a $\frac{1}{2^{L-l}}$, inversamente proporzionale alla larghezza della cella: l'idea infatti è quella di privilegiare, ovvero pesare di più, le corrispondenze trovate nelle celle di dimensioni minori, individuate nei livelli che hanno una risoluzione più fine e quindi un valore di l maggiore; si vuole invece a penalizzare le corrispondenze trovate nelle celle più grandi perchè tendenzialmente coinvolgono feature più dissimili.
Mettendo insieme questi principi in una formulazione unica si ottiene la seguente espressione, che prende il nome di *Pyramid Match Kernel*:

$$k^L(X,Y) = I^L + \sum_{l=0}^{L-1} \frac{1}{2^{L-l}}(I^l - I^{l+1}) = \frac{1}{2^L} I^0 + \sum_{l=1}^{L} \frac{1}{2^{L-l+1}} I^l \tag{4.8}$$

Sia il pyramid match kernel che l'intersezione di istogrammi risultano essere dei kernel di Mercer (vedi la sezione 2.6.2).

Dal Pyramid Match Kernel allo Spatial Pyramid Matching

Il Pyramid Match Kernel visto nel paragrafo precedente opera su una rappresentazione dell'immagine *orderless*, che cioè non tiene conto delle informazioni spaziali delle feature associate ai due spazi da confrontare. L'approccio di SPM è invece definito dagli stessi autori di tipo ortogonale ai metodi classici: opera secondo la logica del Pyramid Matching nello spazio 2D dell'immagine, ma adotta le tecniche classiche di clustering nello spazio delle feature. In dettaglio tutte le feature vengono quantizzate in M tipi discreti, secondo l'assunzione che due feature sono in corrispondenza se sono dello stesso tipo, ovvero se vengono quantizzate nello stesso elemento. Ogni canale m fornisce due set di vettori bidimensionali, X_m ed Y_m, contenenti le coordinate delle feature di tipo m trovate in ciascuna immagine. Il kernel finale è dato dalla somma dei kernel calcolati su ciascun canale m, ovvero:

$$K^L(X,Y) = \sum_{m=1}^{M} k^L(X_m, Y_m) \tag{4.9}$$

Questo metodo coincide con l'approccio classico bag of words per $L = 0$, ovvero quando le immagini non vengono suddivise in celle. Poichè il Pyramid

Match kernel è una somma pesata di intersezioni di istogrammi e poichè il numero di feature in una cella è sempre un numero positivo (per cui vale $c \cdot min(a, b) = min(ca, cb)$), è possibile implementare K^L come un singolo istogramma ottenuto dalla concatenazione dei singoli istogrammi di canale, per tutte le risoluzioni, opportunamente pesati. Questo significa che per L livelli ed M canali il vettore risultante ha dimensione

$$M \sum_{l=0}^{L} 4^l = M \frac{1}{3}(4^{L+1} - 1)$$

Ad esempio con un vocabolario visuale di rispettivamente $M = 400$ parole e $L = 3$ livelli si ottengono dei vettori di dimensione 34000.
La figura 4.4 mostra un esempio di descrizione di un'immagine secondo la struttura piramidale discussa, su un vocabolario di $M = 3$ parole visuali rappresentate con simboli grafici diversi. Nell'immagine si noti inizialmente la divisione spaziale per tre livelli di risoluzione diversi; in seguito si noti il conteggio, per ogni canale M e per ogni livello di risoluzione, delle feature che cadono in ciascun bin spaziale; infine ogni istogramma spaziale viene pesato secondo l'equazione 4.8.

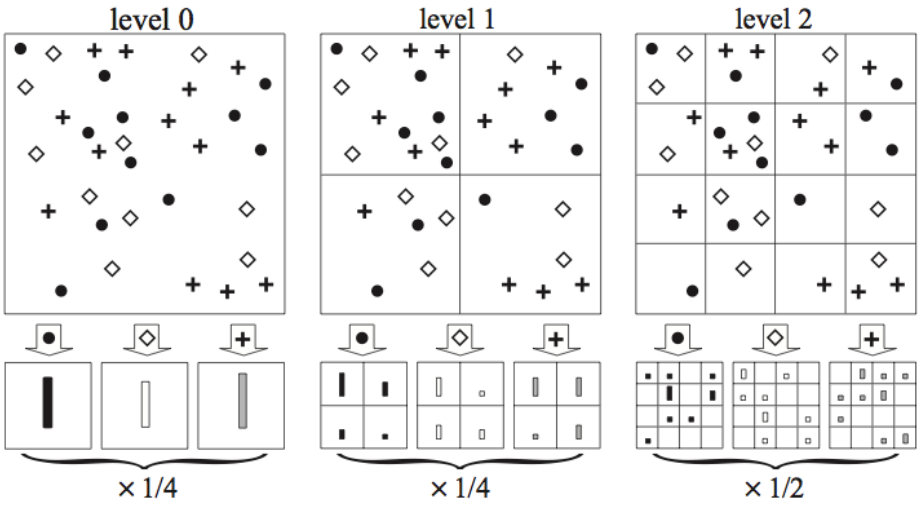


Figura 4.4. Esempio di Spatial Matching Scheme per M=3 ed L=3

BoW piramidale

La descrizione appena vista del modello SPM evidenzia come, a partire da un vocabolario visuale di dimensione $M = N$ termini, si possa ottenere una descrizione dell'immagine rappresentata su una struttura piramidale che assegna un peso maggiore alle quantizzazioni, ovvero i matching tra il vocabolario visuale ed i punti salienti delle immagini, che avvengono ai livelli di risoluzione più fine.
La pipeline che realizza una classificazione basata su SPM, illustrata in figura 4.5, condivide pertanto con il modello bag of words le fasi di raccolta delle feature e di calcolo del dizionario visuale N-dimensionale, mentre si differenzia sostanzialmente per il numero di istogrammi di dimensione N che vengono concatenati nel descrittore delle immagini fornito come input ai metodi di classificazione.
In particolare, fissata una coppia $<$ *tipo di descrittore*, *numero di scale* $>$ e scegliendo una profondità di rappresentazione piramidale pari ad $L = 2$ con un vocabolario di dimensione $N = 200$, la descrizione di ciascuna immagine risulta avere dimensione pari a:

- **3200** se si concatenano nel descrittore finale soltanto i 16 istogrammi N-dimensionali corrispondenti ai nodi della piramide a livello $L = 2$;
- **4200** se nel descrittore finale viene inserita la piramide completa, ovvero i $21 = 1 + 4 + 16$ vettori N-dimensionali che corrispondono a i nodi presenti rispettivamente ai livelli $L = 0, 1, 2$.

Vale la pena far notare che i vettori di intersezione degli istogrammi risultano notevolmente sparsi, visto che un valore alto di L porta ad avere molte celle in cui non si registrano match. Tale caratteristica, così come evidenziato da Grauman e Darrel, rende il carico computazionale di complessità lineare nel numero delle feature. I vettori risultanti dal criterio di descrizione che si decide di applicare sono forniti come input ai metodi di classificazione NN ed SVM nelle modalità descritte in precedenza.

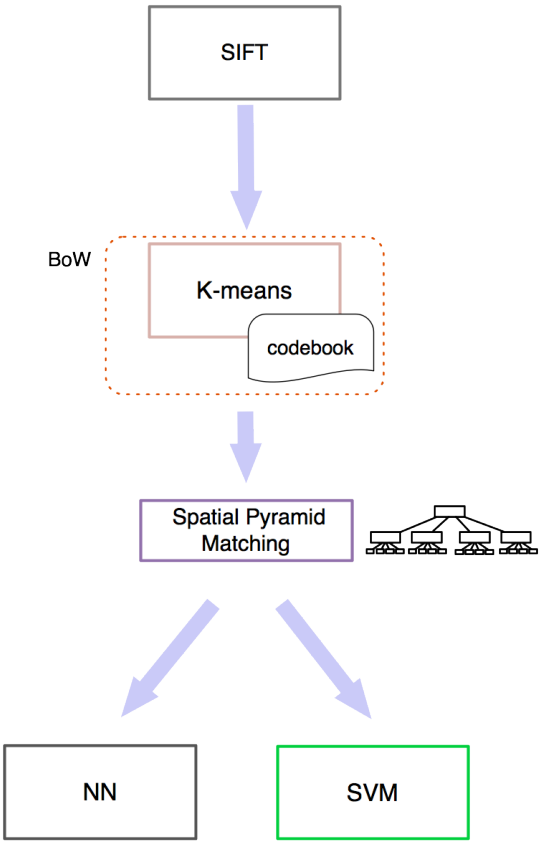


Figura 4.5. Pipeline di classificazione per la modalità di descrizione basata su SPM

4.2.4 GIST

I metodi di descrizione fin qui illustrati sono tutti incentrati sull'analisi di feature locali calcolate sui punti salienti presenti nelle immagini. La pipeline mostrata in questo paragrafo (fig.4.6) viene invece definita su un'analisi globale delle immagini realizzata per mezzo del descrittore GIST [31].

L'idea su cui si basa questo tipo di descrittore è in controtendenza con l'approccio classico adottato nella classificazione delle scene, nel quale si tende ad identificare gli oggetti come le entità fondamentali per il riconoscimento e la corretta interpretazione del contenuto delle immagini.

Il punto di vista alternativo presentato dagli autori consiste nell'inferire la

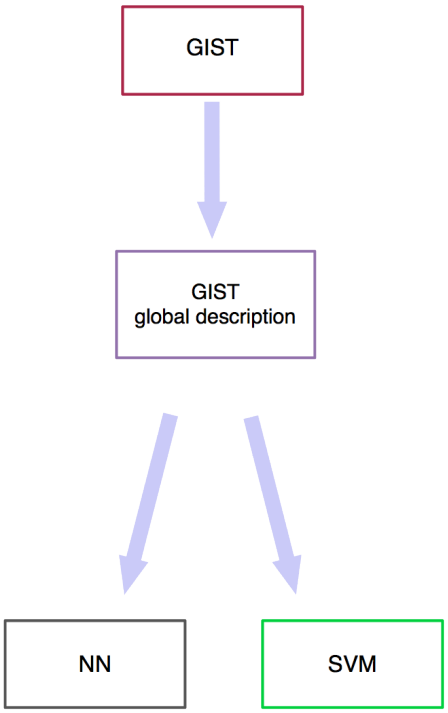


Figura 4.6. Pipeline di classificazione per la modalità di descrizione basata su GIST

categoria semantica di una scena a partire dal suo layout spaziale, facendo precedere pertanto la stima della struttura globale e delle relazioni spaziali tra le componenti della scena all'analisi dei dettagli locali dell'immagine. Da un punto di vista percettivo è stato dimostrato che osservando immagini riprodotte a bassa frequenza si riesce generalmente a percepire l'essenza, chiamata anche *gist*, del loro contenuto. Inoltre alcuni test riguardanti l'interpretazione visuale di immagini caratterizzate da componenti a bassa ed alta frequenza hanno mostrato come, soprattutto per tempi di esposizione all'immagine molto brevi, il contenuto a bassa frequenza spaziale venga talvolta preferito a quello ad alta frequenza per le operazioni di classificazione. Inoltre è stato visto come nelle immagini naturali, a differenza per esempio delle immagini ibride mostrate in figura 4.7, si riscontra una certa contiguità nello spazio scala da parte degli oggetti che compongono la scena. I contorni

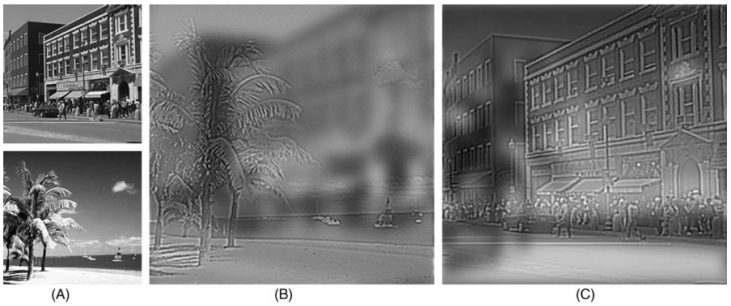


Figura 4.7. Sovrapposizione di due immagini differenti (A) in due combinazioni ibride: in (B) l'immagine della spiaggia viene resa in alta frequenza spaziale mentre la scena urbana in bassa frequenza; in (C) la spiaggia è resa in bassa frequenza spaziale mentre la scena urbana in alta frequenza.

che esistono a basse frequenze spaziali, ad esempio, esisteranno anche ad alte frequenze mentre contorni adiacenti tendono statisticamente ad avere orientazioni simili: queste proprietà rendono possibile per un sistema visivo una rapida costruzione della struttura dell'immagine.
Mettendo insieme quindi le attività di detector di feature a basso livello su larghe regioni del campo visivo è possibile ricostruire la struttura di una scena senza ricorrere alla segmentazione oppure alla ricerca di oggetti e feature locali, con evidente vantaggio in termini computazionali.

Una naturale estensione di questo principio consiste nell'utilizzo non di un'unica feature globale, ma di una collezione di feature globali in grado di catturare la variabilità dei layout e dei punti di osservazione che una scena presenta nel mondo reale.
Il descrittore GIST racchiude un set di feature globali scelte per l'analisi delle scene, le quali vengono calcolate sulla base dell'output ottenuto attraverso la convoluzione delle immagini con un insieme di filtri orientati e multiscala. Scelti un numero N di sample del filtro per ciascuna dimensione ed un valore K di orientazioni e scale, la dimensione del descrittore per un'immagine in scala di grigi diventa $N \times N \times K$, dove $N \times N$ è l'insieme dei valori in cui viene salvata la risposta del filtro per ciascun valore di scala ed orientazione.

Questo tipo di analisi permette di descrivere alcune proprietà che costituiscono lo *Spatial Envelope* della scena, come ad esempio *naturalness, openness* o *roughness*, che si rivelano decisive per l'interpretazione semantica dell'immagine al pari delle metodologie orientate al riconoscimento degli oggetti che popolano la scena.
Data una collezione di immagini, il descrittore finale applicato a ciascun documento è ottenuto dall'esecuzione dell'algoritmo che individua le feature globali per esso: dunque la pipeline che esegue GIST non necessita di operazioni di clustering o quantizzazione, che invece erano necessarie per codificare in un vocabolario i set di descrittori puntuali.
Supponendo di usare 20 valori differenti di orientazioni e scale con una risposta a ciascuna filtro data da un vettore di 16 elementi su 3 canali di colore, la dimensione risultante per il descrittore finale di un'immagine diventa $D = 20 \times 16 \times 3 = 960$. Ciascun vettore $D - dimensionale$ viene utilizzato per le fasi di classificazione NN ed SVM nelle modalità viste in precedenza, ovvero il vettore descrittore GIST di un'immagine di test viene usato per trovare i k vettori più vicini nello spazio N-dimensionale delle immagini di training oppure è usato per valutare il suo grado di similarità nelle matrici di kernel create dai metodi SVM.

4.2.5 Analisi dei tag

La letteratura riguardante la classificazione di immagini si è cimentata negli ultimi anni con il problema di interpretare la grande quantità di dati che confluiscono continuamente su Internet e che provengono dall'attività quotidiana degli utenti di tutto il mondo. Le immagini pubblicate su social network come Facebook o Flickr sono generalmente corredate da un insieme di tag o metadati; questo dato di fatto suggerisce di estendere la comune analisi delle immagini con l'integrazione di tecniche per l'analisi del contenuto testuale. In realtà la classificazione di documenti testuali, come visto nel paragrafo 2.1, fornisce i principi di base su cui si fondano le principali tecniche di classificazione di immagini; ciò che risulta nuovo è però la possibilità di combinare il contenuto visivo e testuale di uno stesso documento in un unico sistema di

classificazione. Per questo scopo, oltre al dataset di riferimento su cui sono stati condotti gli esperimenti, ovvero Caltech101, è stato creato un nuovo dataset chiamato **MICC-Flickr-101** di immagini provenienti da Flickr ed appartenenti alle stesse classi del primo. Il nuovo dataset ha la caratteristica di avere, per ciascuna immagine, un insieme di tag inseriti dall'autore del documento.
L'insieme dei tag di un'immagine viene considerato come una feature al pari delle altre descritte finora: il sistema prevede, pertanto, una pipeline unimodale per la classificazione di immagini descritte attraverso l'insieme dei tag a cui sono associate.
La classificazione di documenti testuali, ovvero gli insiemi di tag, è stata realizzata con alcune delle tecniche già descritte in questa sezione: bag of words ed LDA . La figura 4.8 mostra la pipeline che realizza la classificazione mediante modello LDA costruito sull'insieme dei tag. Anche in questo caso il modello a topic latenti, del tutto identico a livello formale a quello usato per l'analisi delle immagini, presuppone che ciascun documento sia descritto da un vettore di *term frequency* calcolato su un vocabolario di parole.
I termini del vocabolario vengono selezionati nell'intero set di tag relativi ai documenti di training; soltanto i termini che sono presenti nel training set oltre una certa soglia di frequenza entrano a far parte del dizionario.
Prima di creare il dizionario è necessario eseguire alcune operazioni di filtraggio dei termini, in particolare:

- **Rimozione delle *stopwords*** Questa operazione consente di eliminare alcune parole che compaiono molto frequentemente, come ad esempio articoli e preposizioni, ma che non forniscono informazioni utili per discriminare gli argomenti trattati nel testo. In particolare per gli esperimenti su MICC-Flickr-101 è stato creato un elenco di *stopwords* appartenenti a 4 lingue diverse (inglese, italiano, francese, spagnolo).

- **Stemming** Questa operazione consiste nella riduzione di un termine dalla sua forma estesa ad un forma flessa o *radice*, che non corrisponde necessariamente alla sua radice morfologica: in questo modo è possibile

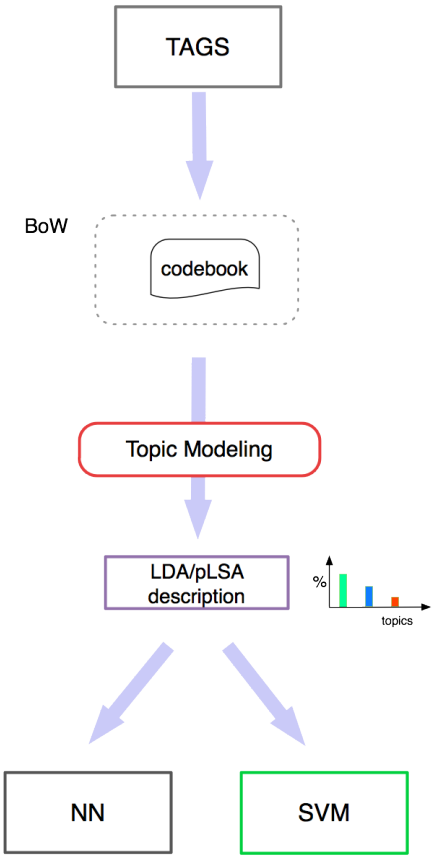


Figura 4.8. Pipeline unimodale che realizza la classificazione attraverso un modello LDA costruito sui tag

mappare forme di parole diverse di concetti simili, come ad esempio i termini *volare, volerò* nella forma unica *vol*.

Una volta ottenuto il dizionario dei termini, i documenti possono essere descritti direttamente dai vettori delle frequenze dei termini, secondo la metodologia classica di bag of words, oppure possono essere descritti dal vettore dei topic generato da un modello LDA.
I descrittori finali dei documenti della collezione vengono infine forniti in input alle modalità NN ed SVM che classificano le immagini di test secondo gli stessi criteri visti per le altre pipeline unimodali.

4.3 Multiple Kernel Learning

Nei paragrafi precedenti è stata descritta la metodologia con cui ciascuna pipeline unimodale utilizzata per gli esperimenti implementa un sistema per la classificazione di una collezione di immagini; ogni pipeline esegue una determinata sequenza di operazioni a partire dalla scelta di una specifica feature per la descrizione dei documenti.
Ogni feature ricerca all'interno delle immagini informazioni di natura diversa, si pensi ad esempio alla differenza tra SIFT e GIST; inoltre feature che sintetizzano informazioni simili possono descrivere i documenti in forme diverse, come ad esempio succede con SIFT e Spatial Pyramid Matching. Infine, a partire da informazioni dello stesso tipo, è possibile descrivere i documenti con un grado di astrazione semantica più elevato, come succede ad esempio nei metodi basati sull'analisi dei topic latenti.
Le metodologie proposte presentano quindi una certa eterogeneità nel modo in cui svolgono l'analisi e la descrizione dei dati. Come logica conseguenza, risulta interessante vedere come la loro *complementarità* possa essere sfruttata in pipeline *multimodali* di classificazione che implementino congiuntamente due o più pipeline unimodali.

Le fusioni di descrizioni di tipo diverso possono avvenire a livello di descrizione dei documenti oppure a livello di calcolo dei kernel a seconda del tipo di feature che si decide di combinare. Nel caso in cui si disponesse di descrittori puntuali come SIFT e di detector di regioni massimamente stabili come MSER[1] sarebbe possibile fondere questi due approcci già a livello di raccolta delle feature, ad esempio limitando la scelta dei punti salienti ai soli elementi che ricadono all'interno di una delle regioni individuate: risulta chiaro che il tipo di fusione così effettuato è possibile nella misura in cui un descrittore, o come nel caso di MSER un detector, possa essere usato come criterio di selezione per un secondo descrittore, come dimostra l'esempio ap-

[1]Nell'analisi di immagini le regioni MSER, *Maximally Stable Extremal Region*, godono della proprietà di essere invarianti a trasformazioni affini ed a variazioni monotone dei valori di intensità.

pena riportato.
La combinazione di feature diverse può invece avvenire a livello di kernel, ovvero dopo che la collezione di dati è stata analizzata e descritta da ciascuna modalità basata su singola feature in maniera indipendente: questa tecnica di fusione, vicina agli approcci di tipo *late fusion*, viene implementata dal **Multiple Kernel Learning**[32].

Il funzionamento di un sistema che realizza MKL si basa sulla formalizzazione del problema della combinazione dei kernel ottenuti dall'addestramento su feature diverse.
Dato un set di F di feature, la similarità tra due elementi x ed x' della collezione valutata sulla m-esima feature f_m viene data dalla funzione kernel $k_m(x, x')$ definita su una qualche metrica di distanza:

$$k_m(x, x') = k(f_m(x), f_m(x')) \tag{4.10}$$

Come abbiamo visto nei precedenti paragrafi, la riga delle matrici kernel $K_m(x)$ per la feature m relativa al generico elemento x, sia per $K_{m_{train}}$ che per $K_{m_{test}}$, ha la forma seguente:

$$K_m(x) = [k_m(x, x_1), k_m(x, x_2), \ldots, k_m(x, x_{T_{train}})] \tag{4.11}$$

Nel caso in cui la descrizione del dataset sia stata fatta raccogliendo F feature diverse, l'apprendimento con SVM sull'intero set avviene utilizzando matrici di kernel ottenute come combinazione delle matrici $K_{m_{train}}$ e $K_{m_{test}}$ definite su ciascuna delle feature usate.

I metodi MKL utilizzati per gli esperimenti corrispondono alle combinazioni *baseline* proposte in [32]; nel dettaglio sono stati utilizzate le seguenti combinazioni tra matrici di kernel:

- **Average** Il metodo più semplice di combinazione tra kernel consiste nel calcolarne la media; la funzione di combinazione k^* risultante diventa:

$$k^*(x, x') = \frac{1}{F} \sum_{m=1}^{F} k_m(x, x') \tag{4.12}$$

- **Product** La seconda combinazione *baseline* considerata consiste nella moltiplicazione tra kernel; in questo caso la funzione k^* risultante è la seguente:

$$k^*(x, x') = (\prod_{m=1}^{F} k_m(x, x'))^{\frac{1}{F}} \tag{4.13}$$

Per ciascuno di questi metodi la classificazione delle immagini di test prevede l'uso di un parametro di costo C opportunamente valutato attraverso un processo di cross-validation.
I metodi *baseline* appena descritti combinano l'insieme dei kernel in maniera deterministica; una naturale evoluzione di questo approccio consiste in realtà nell'assegnare alle funzioni $k_m(x, x')$ coinvolte dei pesi $\beta_m \geqslant 0$ calcolati durante la fase di training e tali che $\sum_{m=1}^{F} \beta_m = 1$. Il motivo per cui gli esperimenti sono stati condotti sulle modalità *baseline* e non sui metodi MKL più complessi è stato dettato dal fatto che, come si evince in [32], le due combinazioni in esame forniscono dei risultati paragonabili ed in alcuni casi superiori ad altri metodi più raffinati per lo stesso dataset e su feature simili a quelle utilizzate in questo lavoro.

Nella figura 4.9 viene illustrata una delle combinazioni tra pipeline unimodali implementate negli esperimenti di MKL. Lo schema in esame mostra in particolare la combinazione tra una prima pipeline che descrive il dataset attraverso un modello LDA, una seconda che descrive attraverso uno schema SPM ed una terza che analizza la collezione con il descrittore GIST: si noti come, a livello concettuale, una configurazione di questo tipo consenta di combinare in una rappresentazione comune l'efficacia discriminativa di 3 approcci diversi.
L'esecuzione della pipeline multimodale presuppone l'esecuzione indipendente delle pipeline unimodali coinvolte fino al calcolo delle matrici di kernel: per ciascuna istanza di pipeline unimodale si avrà un numero di matrici pari alle funzioni di kernel descritte in 4.1.2; nel passo successivo le matrici vengono ricombinate secondo le funzioni di media e di prodotto e fornite ad un modulo SVM che le userà per la fase di training del modello e di classificazione delle immagini di test. Le n matrici scelte per combinare n pipeline unimodali

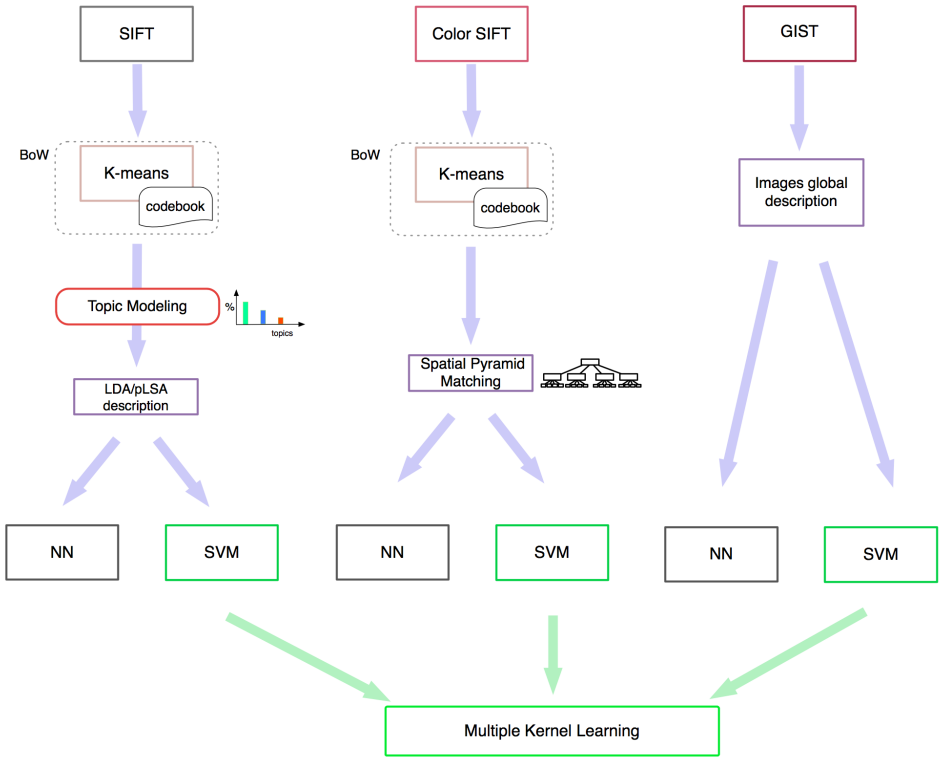


Figura 4.9. Esempio di MKL ottenuto combinando le pipeline unimodali costruite su LDA, SPM e GIST

provengono ciascuna da una pipeline diversa ed in nessun caso ci si troverà ad avere più di una matrice kernel per una determinata feature: questo vincolo garantisce che le combinazioni calcolate siano sempre consistenti con l'idea di sperimentare fusioni di modalità complementari. Supponendo quindi di calcolare k matrici diverse per ciascuna delle n pipeline unimodali e di scegliere di volta in volta una sola matrice per ciascuna modalità, si avranno in totale k^n combinazioni diverse per ciascuna funzione *baseline* utilizzata per fare MKL.

Capitolo 5

Esperimenti

In questo capitolo vengono fornite in maniera dettagliata le configurazioni ed i risultati ottenuti per gli esperimenti di classificazione. I test sono stati condotti su due dataset di riferimento, Caltech-101 e MICC-Flickr-101, per ciascuno dei quali viene proposta una breve descrizione prima di fornire i risultati numerici.

5.1 Caltech-101

Il dataset Caltech-101 è stato realizzato da Fei-Fei Li, Marco Andreetto, e Marc Aurelio Ranzato, ricercatori di visione computazionale del California Institute of Technology, nel 2003. È costituito da immagini di oggetti appartenenti a 101 categorie diverse più una classe di immagini di "background" ottenuta cercando la parola "things" su Google Immagini. Il numero di immagini per categoria varia da 40 a 800, con una media di 50 cadauna; le dimensioni di ciascuna immagine sono fissate intorno a 300x200 pixel.

Il dataset è stato introdotto per la prima volta in un lavoro presentato al CVPR del 2004 [12]; una versione estesa è stata pubblicata nel 2006 [13]. In questo secondo lavoro viene presentato un sistema di classificazione caratterizzato da un metodo probabilistico generativo di tipo bayesiano basato su un modello *constellation* di rappresentazione delle feature: ogni immagine viene rappresentata da un insieme di parti, ognuna delle quali codifica in-

Figura 5.1. Le categorie del dataset Caltech-101: per ognuna di esse sono mostrate due immagini casuali prese dall'insieme complessivo. L'ultima riga mostra esempi dal dataset di background.

formazioni sia sulla propria forma che sulla posizione reciproca rispetto alle altre. La selezione delle 101 categorie che compongono il dataset si è basata su una ricerca all'interno Webster Collegiate Dictionary [1], scegliendo un sottoinsieme dei termini che fossero associati ad un disegno. La lista dei nomi delle categorie così generata è stata usata per scaricare immagini tramite Google Image Search; questo primo insieme è stato poi filtrato manualmente per eliminare immagini inconsistenti. Le immagini scelte sono state sottoposte ad ulteriori passi di elaborazione: quando necessario sono state applicate trasformazioni di ribaltamento e/o rotazione per fare in modo che tutte le istanze di una stessa classe mostrassero oggetti orientati lungo la stessa direzione. Al termine di questa fase di preprocessing tutte le immagini sono state scalate in modo da impostare la dimensione massima su 300 pixel. L'elenco delle categorie ed alcuni esempi di immagini sono mostrati in figura 5.1.
L'obiettivo del lavoro presentato da Fei-Fei Li è mostrare che un modello accurato per la classificazione può essere ottenuto a partire da un numero molto ristretto di esempi di training. Per ogni categoria vengono scelte N immagini per l'addestramento in modo casuale, con $N = \{0, 1, 3, 6, 15\}$; i test sono eseguiti con al più altre 50 immagini per categoria scelte sempre in modo casuale tra le rimanenti. Se una classe non dispone di 50 immagini per il test set, vengono utilizzate tutte quelle disponibili. Gli esperimenti vengono ripetuti 10 volte, scegliendo ogni volta nuovi insiemi casuali di immagini. Il caso di $N = 0$ prevede che si usi per la classificazione soltanto il modello a priori del metodo bayesiano.
Il setup degli esperimenti è stato riproposto in modo simile anche in altri lavori che usano Caltech-101 per i test. Ad esempio gli autori di [15] propongono di scegliere 30 immagini in modo casuale da ogni classe del dataset, scelta condivisa anche nel testare il metodo Spatial Pyramid Matching [37]. Anche il metodo introdotto in [7] è stato valutato su Caltech-101 variando il numero di immagini di addestramento per classe tra 15 e 30 ed usandone al più 50 per il test, ripetendo il processo 10 volte. Altri metodi successivi [41] [23] [43] hanno mantenuto queste impostazioni.

5.1.1 Setup degli esperimenti

Il primo passo per l'esecuzione di esperimenti di classificazione sul dataset Caltech-101 è stato quello di fissare alcuni parametri liberi. Anzitutto, in accordo a quanto osservato nel paragrafo precedente, si è stabilito di eseguire le prove utilizzando 30 immagini scelte a caso da ogni classe per formare il training set e al più 50 per classe, o in base al numero massimo, per il test set. Visti i prolungati tempi di esecuzione, che in alcuni casi si sono protratti anche per intere giornate per eseguire una singola istanza di pipeline unimodale, gli esperimenti sono stati ripetuti solo 3 volte invece di 10; il numero di ripetizioni è in ogni caso sembrato sufficiente per mostrare risultati che derivassero da una certa variabilità nell'insieme di train e di test.
I parametri inerenti al tipo di descrittori ed alle relative variabili di dimensionalità sono stati fissati in modo da poter ottenere risultati che fossero comparabili con i principali articoli di riferimento del lavoro presentato, in particolare rispetto al lavoro di Bosch [7] e a quello di Lazebnik [37]. I descrittori usati sono stati il SIFT classico, estratti con campionamento denso a scala singola e a quattro scale, ed una variante del SIFT a colori, RGBSIFT, a quattro scale. Nella variante a singola scala essa si suppone fissata al valore unitario, intendendo in questo modo di avere dei descrittori SIFT calcolati su una finestra di 16x16 pixel. Applicando questa convenzione, le varianti multiscala sono state calcolate alle scale 0.5, 1, 1.5, 2; tutti i punti derivano da un campionamento denso su una griglia di passo pari ad 8 pixel. I test sul modello bag of words classico sono stati eseguiti facendo variare il numero di parole tra 200 e 1500; quelli sui modelli a topic latenti con un numero di topic variabile tra 70 e 200, a partire da modelli BoW di 500 e 1000 parole per SIFT a singola scala, 1000 e 1500 parole per SIFT ed RGBSIFT a quattro scale.

Esperimenti preliminari

Per la scelta del descrittore SIFT a colori con cui eseguire gli esperimenti sono stati messi a confronto i metodi C-SIFT, OpponentSIFT, HueSIFT ed

	NN-L2	**SVM-l**	**SVM-C2**	**SVM-I**
HueSIFT	15.2 ± 1	22.4 ± 0.8	28.2 ± 1.3	28.1 ± 1.4
C-SIFT	20.1 ± 1.5	30.6 ± 1.5	34.5 ± 1.1	33.5 ± 0.9
OppSIFT	23.2 ± 1.3	33.2 ± 1.1	38.6 ± 1.5	38.5 ± 1.3
RGBSIFT	23.6 ± 0.8	33.6 ± 0.9	39.9 ± 1	39.7 ± 1.1

Tabella 5.1. Risultati degli esperimenti di classificazione con diverse varianti di SIFT a colori

RGBSIFT, come descritti nel capitolo 2 e nell'articolo di riferimento [39]. Per l'estrazione di questi descrittori è stato usato il software messo a disposizione dall'autore dell'articolo. I test sono stati eseguiti con descrittori estratti su una griglia a passo 10 su tre scale (0.5, 1, 1.5) e impostando la dimensione del codebook a 200 parole, ripetendo il processo di creazione del dizionario per 10 scelte casuali di immagini all'interno del dataset, che sono state mantenute costanti durante tutti i test. I risultati ottenuti sono proposti in tabella 5.1, riportando i valori con la notazione *media* ± *varianza*. Dai test eseguiti risulta che le migliori prestazioni di classificazione sono state registrate usando RGBSIFT; questo metodo è stato perciò scelto come soluzione color-SIFT per tutti i test successivi.

Sono state testate le performance di classificazione dei metodi k-NN e k-NN-class al variare di k per trovare il valore ottimo in entrambi i casi e per entrambe le metriche usate, L2 e χ^2. I test sono stati eseguiti utilizzando il descrittore SIFT a scala singola, calcolato per una griglia di punti a passo 8, impostando la dimensione del vocabolario a 500 e riducendo la descrizione a 60 topic tramite LDA. I valori di k sono stati fatti variare tra 1 e 30, valore massimo del numero di vicini per classe per il metodo NN-class. I risultati sono mostrati in figura 5.2. Come si vede il metodo k-NN-class presenta un picco nelle prestazioni di classificazione impostando $k = 3$, mentre per il k-NN classico esse non migliorano di molto all'incremento del valore del parametro. Di conseguenza per il primo metodo si è scelto di utilizzare $k = 1$, per il metodo basato sulla distanza di classe invece $k = 3$.

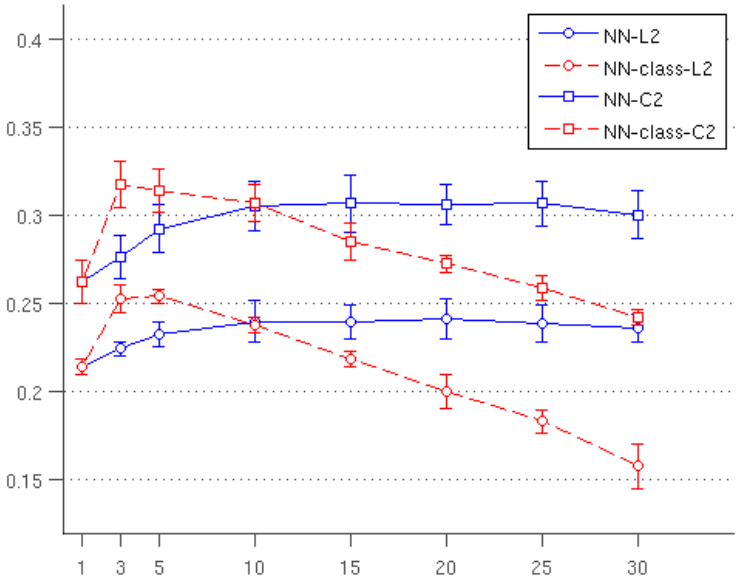


Figura 5.2. Test eseguiti sui metodi k-NN e k-NN-class al variare di k

5.1.2 Bag of Words

Per ottenere informazioni sulle prestazioni dell'algoritmo bag of words di base sono stati eseguiti numerosi test variando la dimensione W del vocabolario visuale per i metodi SIFT a singola scala, SIFT multiscala ed RGBSIFT multiscala, fissando i parametri liberi come specificato nel paragrafo precedente. I tempi di esecuzione per un singolo esperimento comprensivo del calcolo dell'accuratezza per tutte le modalità, esclusi i tempi di creazione del dizionario, è stato variabile tra 20 minuti per il SIFT a scala singola con codebook di $W = 200$ parole e circa 3 ore usando RGBSIFT con codebook $W = 1500$. In totale sono state necessarie 40 ore per completare i test su SIFT a singola scala, 55 ore per l'analogo multiscala e 65 ore per RGBSIFT multiscala. Essendo stati questi i primi test eseguiti durante il lavoro di tesi, non per tutti i casi sono stati sperimentati tutti i metodi proposti; nello specifico NN-χ^2 ed SVM-$exp\chi^2$ sono stati introdotti solo per i metodi multiscala mentre NN-class è stato introdotto soltanto nelle pipeline successive.
Analizzando i risultati ottenuti, riassunti in figura 5.3, si nota come per i

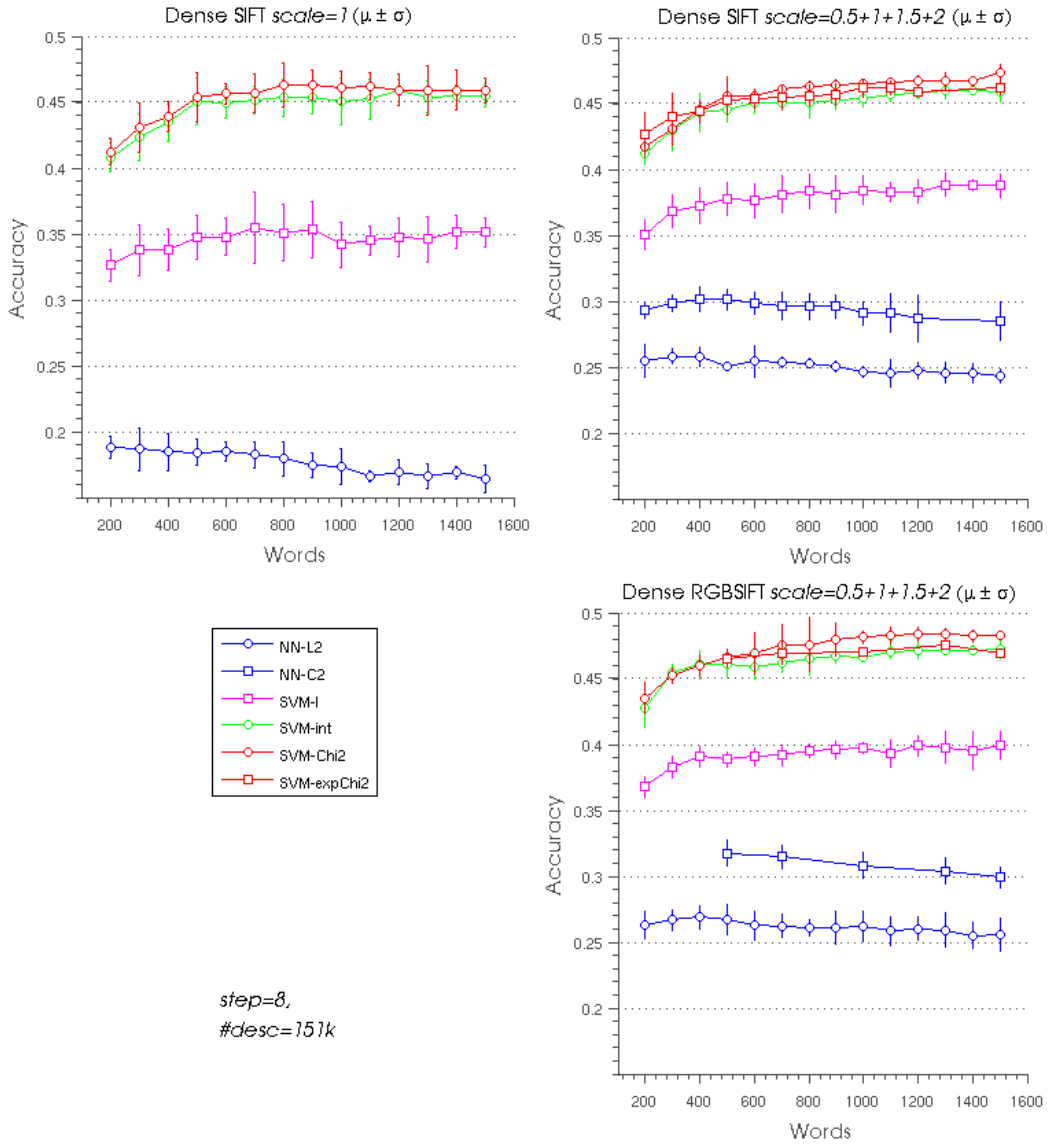


Figura 5.3. Risultati baseline metodi BoW; in senso orario partendo da in alto a sinistra: SIFT a singola scala, SIFT a 4 scale, RGBSIFT a 4 scale

metodi NN le prestazioni massime sono variabili tra il 18% e il 32%: in particolare risultano migliori per piccoli valori di W, subendo una riduzione di circa il 2% al crescere del suo valore. Si può inoltre rilevare un notevole incremento, quantificabile intorno al 7%, aumentando il numero di scale, ovvero il numero di descrittori calcolati in ogni immagine. Tra i metodi NN quello che consente di ottenere i valori migliori è NN-χ^2. I metodi SVM raggiungono prestazioni di accuratezza massime variabili tra il 45% e il 48%, ottenute utilizzando il kernel χ^2 o $exp(\chi^2)$; la variazione della dimensione del vocabolario può portare ad un incremento delle prestazioni fino al 5%, mentre l'incremento delle scale nella fase di campionamento migliora le prestazioni di circa l'1%.

5.1.3 Modelli a topic latenti

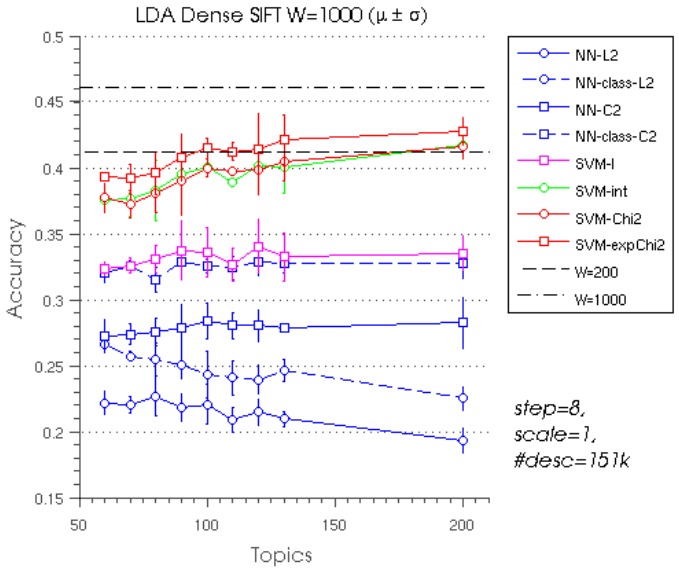


Figura 5.4. Risultati LDA SIFT (singola scala, $W = 1000$)

Il dettaglio dei test eseguiti su LDA è riportato nelle figure 5.4, 5.5 e 5.6. Per verificare le prestazioni dell'implementazione scelta e confrontarle col metodo BoW classico sono stati fissati tre valori della dimensione del dizionario, $W = \{500, 1000, 1500\}$. Per ognuno di essi sono stati eseguiti dei test per va-

lori del numero di topic nell'insieme $Z = \{70, 80, 90, 100, 110, 120, 130, 200\}$. I valori più piccoli di Z sono stati introdotti per avere dei dati comparabili con la letteratura; il valore $Z = 200$ per confrontare il modello a topic latenti con le prestazioni ottenute con una pipeline BoW impostando $W = 200$. In ogni figura pertanto sono stati riportati come riferimento i valori di prestazioni ottenute con due modelli BoW, di cui il primo calcolato sul codebook da cui deriva il modello LDA dell'esperimento, mentre il secondo calcolato su una dimensione di riferimento pari a $W = 200$.

Il principale vantaggio che deriva dall'uso dei metodi a topic latenti è la

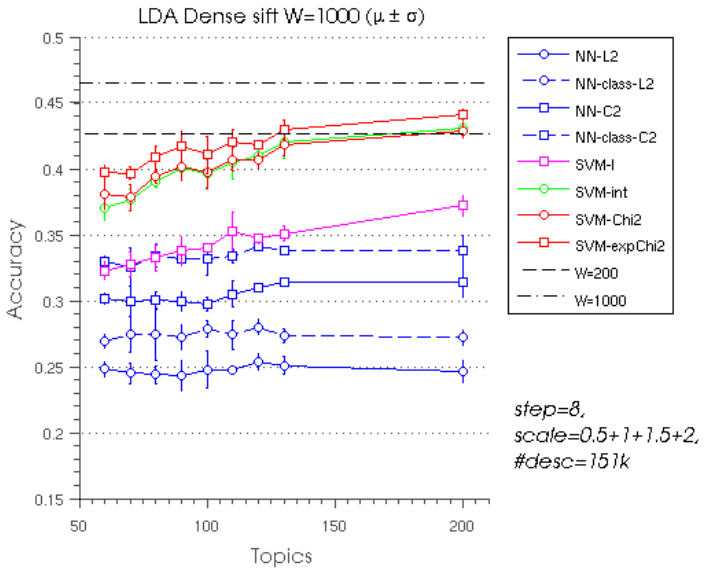


Figura 5.5. Risultati LDA SIFT (quattro scale, $W = 1000$)

riduzione della dimensione della descrizione dell'immagine che comporta una notevole diminuzione nei tempi di esecuzione. L'implementazione usata per LDA consente di ottenere le distribuzioni di topic a partire dalle frequenze delle parole con un'elaborazione di pochi minuti. Escludendo comunque i tempi necessari alla creazione del codebook, i tempi di calcolo dipendono soltanto dal numero di topic usato per la rappresentazione ed assumono valori variabili tra gli 8 ($Z = 60$) ed i 14 ($Z = 200$) minuti. In questi test sono stati testati tutti i metodi descritti in questo lavoro, compresi SVM-$exp(\chi^2)$

e NN-class.

I risultati ottenuti per SIFT a singola scala mostrano prestazioni decrescenti per i metodi NN-L2 all'aumentare del numero di topic; i valori crescono invece in modo concorde in caso si usino metodi SVM. I valori massimi di accuratezza di classificazione variano tra il 21% ed il 32% per le modalità NN: da notare l'incremento di prestazioni ottenuto grazie ai metodi NN-class, che arriva in alcuni casi fino al 7%. Per i metodi SVM si raggiungono nei casi migliori valori intorno al 43%. La variazione della dimensione iniziale del codebook porta ad un incremento delle prestazioni trascurabile, pari al più all'1%. Per questo motivo in figura 5.4 viene riportato soltanto il dettaglio dei risultati per $W = 1000$.
Introducendo le modalità multiscala si notano configurazioni analoghe al caso BoW: un incremento di 4-5 punti percentuali per i metodi NN, di solo 1-2% per SVM. Sia la variazione della dimensione del dizionario che l'uso di descrittori a colori portano ad un complessivo incremento delle prestazioni; in entrambi i casi comunque si tratta di un guadagno quantificabile intorno ad un punto percentuale. Nelle figure 5.5 e 5.6 sono riportati i risultati nel dettaglio per le configurazioni che hanno restituito le migliori prestazioni, ovvero rispettivamente per $W = 1000$ e $W = 1500$.

5.1.4 Metodi unimodali

Nelle tabelle 5.2 e 5.3 viene mostrato un riepilogo delle migliori prestazioni ottenute per i metodi BoW e per quelli a topic latenti. Essi vengono confrontati con i risultati derivati dalla sperimentazione degli altri metodi unimodali introdotti nel capitolo precedente: Spatial Pyramid Matching applicato a SIFT ed RGBSIFT, indicato qui con la sigla CSIFT, e GIST. Il riepilogo serve sia per poter confrontare rapidamente i diversi algoritmi che per avere una visione d'insieme delle pipeline unimodali che sono state fuse nella fase di Multiple Kernel Learning. Tutti i risultati sono espressi in valori percentuali.

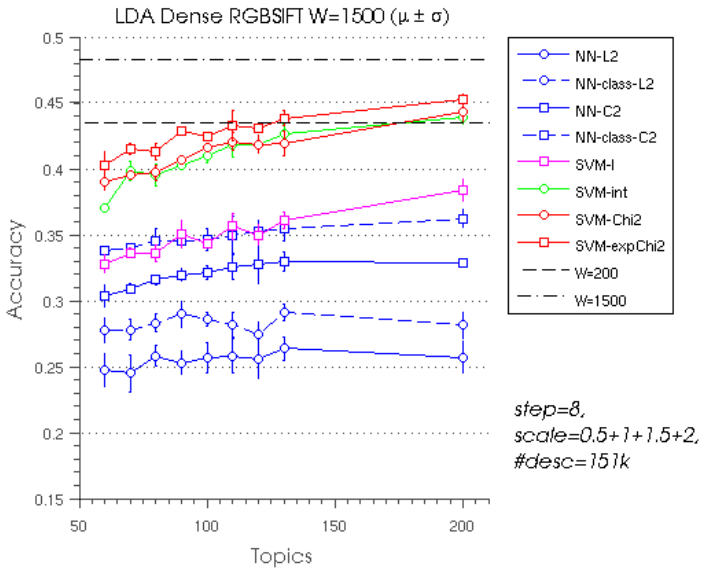


Figura 5.6. Risultati LDA RGBSIFT (quattro scale, $W = 1500$)

I valori riportati per le modalità BoW ed a topic latenti si riferiscono in

	L2	**class-L2**	χ^2	**class** $-\ \chi^2$
$\mathbf{BoW_{SIFT}}$	25.8 ± 0.6	-	30.1 ± 1.0	-
$\mathbf{BoW_{RGBSIFT}}$	26.9 ± 0.9	-	31.8 ± 0.1	-
$\mathbf{LDA_{SIFT}}$	25.2 ± 1.4	27.9 ± 0.6	31.3 ± 05	$\mathbf{34.3 \pm 1.2}$
$\mathbf{LDA_{RGBSIFT}}$	26.4 ± 1.2	29.1 ± 1.0	33.0 ± 0.3	$\mathbf{36.2 \pm 0.7}$
$\mathbf{SP_{SIFT}}$	30.2 ± 1.4	31.6 ± 0.7	39.4 ± 0.8	$\mathbf{40.2 \pm 1.2}$
$\mathbf{SP_{RGBSIFT}}$	30.9 ± 1.6	31.3 ± 1.2	40.0 ± 1.0	$\mathbf{40.9 \pm 1.3}$
GIST	31.4 ± 0.8	$\mathbf{35.6 \pm 0.4}$	33.0 ± 1.0	34.7 ± 0.9

Tabella 5.2. Metodi NN

ogni caso al migliore risultato per metodo registrato al variare dei parametri W e Z. Ad esempio, per SIFT sono stati riportati i risultati ottenuti con $W = 1400 - 1500$ per SVM, $W = 300 - 400$ per NN; per RGBSIFT $W = 1300 - 1500$ per SVM, $W = 400 - 500$ per NN. Per i modelli a topic latenti, $W = 1500$ per tutti i valori riprodotti; $Z = 200$ per SVM, $Z = 110 - 130$

per NN.

I test sulla modalità basata su spatial pyramid matching sono stati eseguiti seguendo le impostazioni in [37]: per un valore di $W = 200$, considerando il caso "strong features", sono stati sperimentati i descrittori SIFT e RGBSIFT multiscala. La piramide è stata calcolata su tre livelli ($L = 2$), ottenendo un descrittore complessivo di dimensione pari a 4200. Il tempo necessario per l'esecuzione di un singolo esperimento varia tra 4 e 5 ore, per un totale di circa 15 ore per l'esecuzione di 3 ripetizioni applicate a ciascuna scelta di descrittore.
Il descrittore GIST utilizzato opera a livello globale e calcola una descrizione dell'immagine su un vettore di 960 elementi. Il tempo di esecuzione di una singola prova è intorno alle 4 ore.

	Lin.	**Int.**	χ^2	**exp**χ^2
$\mathbf{BoW_{SIFT}}$	38.4 ± 1.1	46.0 ± 0.2	$\mathbf{47.3 \pm 0.7}$	46.2 ± 0.5
$\mathbf{BoW_{RGBSIFT}}$	40.0 ± 1.0	47.2 ± 0.7	$\mathbf{48.4 \pm 0.4}$	47.6 ± 0.1
$\mathbf{LDA_{SIFT}}$	37.0 ± 1.7	43.4 ± 2.1	43.0 ± 1.7	$\mathbf{44.3 \pm 1.9}$
$\mathbf{LDA_{RGBSIFT}}$	38.4 ± 0.8	43.9 ± 0.5	44.3 ± 0.7	$\mathbf{45.2 \pm 0.3}$
$\mathbf{SP_{SIFT}}$	49.2 ± 1.0	50.3 ± 0.4	$\mathbf{59.0 \pm 1.2}$	49.9 ± 0.4
$\mathbf{SP_{RGBSIFT}}$	50.2 ± 0.8	51.6 ± 1.6	$\mathbf{59.6 \pm 1.6}$	50.9 ± 1.2
GIST	46.7 ± 1.6	47.6 ± 1.5	$\mathbf{48.7 \pm 1.6}$	48.6 ± 1.2

Tabella 5.3. Metodi SVM

5.1.5 MKL

Gli esperimenti di Multiple Kernel Learning sono stati eseguiti per quantificare il miglioramento apportato in termini di prestazioni dalla combinazione dei kernel ottenuti al termine del processo di analisi delle feature unimodali, cercando di stabilire al contempo l'insieme dei parametri ottimali per ognuna di esse. Particolare attenzione è stata inoltre posta nel confronto tra i risultati ottenuti impiegando metodi BoW classici e quelli raggiunti con l'introduzione

della rappresentazione a topic latenti. Per ogni tipo di feature, e quindi per ogni modello di pipeline, sono stati eseguiti test di fusione con tutti i kernel introdotti in 4.1.2. I metodi di fusione impiegati sono stati *average* e *product*, secondo i criteri descritti nel precedente capitolo. In tutte le configurazioni considerate è stato sempre verificato che fossero utilizzate soltanto feature differenti, evitando cioè test del tipo $BoW_{SIFT\ W=200} - BoW_{SIFT\ W=1000}$. Le descrizioni tramite SIFT ed RGBSIFT, pur appartenendo alla stessa famiglia, sono state considerate come feature differenti e pertanto combinate in alcune configurazioni di MKL.
I primi esperimenti sono stati condotti fondendo tra loro i kernel ottenuti a partire da feature SIFT multiscala, RGBSIFT multiscala e GIST. In particolare si sono testate tutte le possibili combinazioni di 2 o 3 feature, con il parametro W variabile nei due insiemi:

$$W_{SIFT} = \{200, 300, 500, 700, 1000\}$$
$$W_{RBGSIFT} = \{500, 700, 1000, 1300, 1500\}$$

Tali insiemi sono stati scelti a partire dalla totalità degli esperimenti unimodali condotti analizzando le prestazioni ottenute. I risultati migliori sono stati ottenuti con il metodo *product* con fusioni di due kernel del tipo $BoW_{RGBSIFT}\ exp(\chi^2)$ - $GIST\ int$, con prestazioni del 60,9%. In generale le fusioni a 3 feature hanno comunque raggiunto prestazioni intorno al 58% sia per *average* che per *product*; in questi casi utilizzare il kernel $exp(\chi^2)$ per ogni feature sembra dare risultati ottimali.
Nella seconda fase i metodi BoW sono stati sostituiti da LDA, considerando in analogia al caso precedente solo un piccolo sottoinsieme dei risultati unimodali a disposizione. Sia per SIFT che per RGBSIFT sono stati utilizzati gli stessi insiemi di codebook, specificati di seguito; per brevità viene adottata la simbologia $< numwords > - < numtopics >$.

$$\begin{aligned} W = \{ \quad & 500-70,\ 500-100,\ 500-200, \\ & 1000-70, 1000-100, 1000-200, \\ & 1500-70, 1500-100, 1500-200 \quad \} \end{aligned}$$

Le prestazioni massime si sono ottenute per fusioni di coppie di feature del tipo $LDA_{RGBSIFT\ 1500-200}\ exp(\chi^2)$ - $GIST\ exp(\chi^2)$, ottenendo valori intorno al 58% sia per *average* che per *product*. Nelle fusioni a 3 feature i valori di accuratezza non salgono mai oltre il 47%; i valori massimi si ottengono per combinazioni $1500-200$, utilizzando $exp(\chi^2)$ per ogni pipeline unimodale.

Nelle due fasi finali è stato introdotto anche il modello Spatial Pyramid, calcolato sia a partire da feature SIFT che RGBSIFT. In questo modo il numero complessivo di possibili fusioni tra feature è salito a 5. Anche in questo caso sono state eseguite due batterie di test, usando prima metodi BoW ed in seguito i topic latenti. Per velocizzare i test è stato ulteriormente ridotto l'insieme delle possibili combinazioni di dimensione del vocabolario per BoW e numero di topic per LDA, scegliendo soltanto le feature che massimizzavano i test precedenti. In particolare, nel primo caso si sono considerate solo $BoW_{SIFT\ W=1000}$ ed $BoW_{RGBSIFT\ W=1500}$, mentre nel secondo $LDA_{SIFT\ 1000-200}$ ed $LDA_{RGBSIFT\ 1500-200}$. Per semplicità si indicherà nelle tabelle seguenti con BoW_{SIFT} la pipeline $BoW_{SIFT\ W=1000}$, con $BoW_{RGBSIFT}$ la pipeline $BoW_{RGBSIFT\ 1500}$; per i topic latenti analogamente si scriverà soltanto LDA_{SIFT} ed $LDA_{RGBSIFT}$. I migliori risultati ottenuti al variare del numero di feature impiegate per la fusione sono riportati nelle tabelle che seguono.

5.1.6 Valutazione delle prestazioni

Partendo da un'analisi dei primi test eseguiti, si può osservare come le prove su SIFT a singola scala con dimensione del vocabolario $W = 200$ siano in accordo con i risultati riportati in [37]: 41.2% rispetto al nostro 40.6%. L'introduzione di scale multiple e di descrittori a colori porta ad un incremento massimo di prestazioni intorno al 7%.
Osservando i risultati dei test comparativi eseguiti per il modello a topic latenti scelto, si osserva che in caso $Z = W = 200$ le prestazioni di LDA superano quelle del BoW classico del 2%. Le prestazioni migliori si ottengono

	Average		**Product**	
#	value	feats	value	feats
2	$\mathbf{63.0 \pm 0.8}$	**$SP_{RGBSIFT}$ χ^2** **GIST χ^2**	$\mathbf{61.4 \pm 1.6}$	**$SP_{RGBSIFT}$ int** **GIST int**
3	63.1 ± 1.0	$SP_{RGBSIFT}$ χ^2 SP_{SIFT} int $GIST$ χ^2	61.7 ± 1.1	SP_{SIFT} int $SP_{RGBSIFT}$ int $GIST$ int
4	63.0 ± 1.3	$BoW_{RGBSIFT}$ int $SP_{RGBSIFT}$ χ^2 SP_{SIFT} $exp(\chi^2)$ $GIST$ $exp(\chi^2)$	61.3 ± 1.0	$BoW_{RGBSIFT}$ $exp(\chi^2)$ $SP_{RGBSIFT}$ int SP_{SIFT} int $GIST$ $exp(\chi^2)$
5	62.4 ± 1.1	$BoW_{RGBSIFT}$ $exp(\chi^2)$ BoW_{SIFT} int $SP_{RGBSIFT}$ χ^2 SP_{SIFT} χ^2 $GIST$ $exp(\chi^2)$	60.6 ± 0.3	$BoW_{RGBSIFT}$ int BoW_{SIFT} int $SP_{RGBSIFT}$ int SP_{SIFT} int $GIST$ $exp(\chi^2)$

Tabella 5.4. Risultati MKL per fusioni a 2,3,4 e 5 feature utilizzando modelli bag of words

per i metodi multiscala, in corrispondenza di un notevole fattore di riduzione dimensionale ($W = 1500 \rightarrow Z = 200$); proprio in questi casi una riduzione della dimensionalità di un fattore 7.5 porta ad una riduzione di accuratezza di quasi 3% rispetto ai metodi BoW. Le prestazioni migliori in assoluto si ottengono con kernel del tipo SVM-$exp(\chi^2)$; tra i metodi NN quello che consente di ottenere la migliore accuratezza è NN-class-χ^2.
Le prestazioni ottenute col metodo Spatial Pyramid sono comparabili con quelle presentate in [37]: 59.6 ± 1.6 rispetto ad un originale 64.6 ± 0.8. È probabile che la differenza di prestazioni derivi da una diversa scelta delle immagini degli split oppure da un codice più affinato nella stima dei parametri.

In merito ai risultati complessivi al termine dei test su MKL, si osserva

	Average		**Product**	
#	value	feats	value	feats
2	60.0 ± 1.0	LDA_{SIFT} $exp(\chi^2)$ $SP_{RGBSIFT}$ χ^2	58.1 ± 1.4	$LDA_{RGBSIFT}$ $exp(\chi^2)$ $GIST$ $exp(\chi^2)$
3	$\mathbf{63.2 \pm 1.2}$	$\mathbf{LDA_{RGBSIFT}}$ $\mathbf{exp(\chi^2)}$ $\mathbf{SP_{RGBSIFT}}$ $\mathbf{\chi^2}$ **GIST** $\mathbf{exp(\chi^2)}$	$\mathbf{61.4 \pm 1}$	$\mathbf{LDA_{RGBSIFT}}$ $\mathbf{exp(\chi^2)}$ $\mathbf{SP_{RGBSIFT}}$ **int** **GIST** $\mathbf{exp(\chi^2)}$
4	63.2 ± 1.4	$LDA_{RGBSIFT}$ $exp(\chi^2)$ $SP_{RGBSIFT}$ χ^2 SP_{SIFT} $exp(\chi^2)$ $GIST$ $exp(\chi^2)$	61.6 ± 1.3	$LDA_{RGBSIFT}$ $exp(\chi^2)$ $SP_{RGBSIFT}$ int SP_{SIFT} int $GIST$ $exp(\chi^2)$
5	62.8 ± 1.1	$LDA_{RGBSIFT}$ $exp(\chi^2)$ LDA_{SIFT} $exp(\chi^2)$ $SP_{RGBSIFT}$ χ^2 SP_{SIFT} χ^2 $GIST$ $exp(\chi^2)$	60.7 ± 1.1	$LDA_{RGBSIFT}$ $exp(\chi^2)$ LDA_{SIFT} $exp(\chi^2)$ $SP_{RGBSIFT}$ int SP_{SIFT} int $GIST$ $exp(\chi^2)$

Tabella 5.5. Risultati MKL per fusioni a 2,3,4 e 5 features utilizzando modelli a topic latenti

che questo metodo di fusione porta ad un miglioramento massimo del 4% rispetto ai migliori risultati ottenuti con i metodi a piramide spaziale; i metodi BoW vengono migliorati del 15%, i modelli LDA del 18%. Da notare inoltre che l'introduzione di LDA al posto dei modelli BoW comporta un lieve incremento nelle prestazioni finali, a dimostrazione della maggiore complementarità apportata dal livello semantico introdotto dal modello a topic latenti; da notare come tra tutte le pipeline unimodali i metodi di tipo Spatial Pyramid si rivelino sempre vincenti rispetto al bag of words classico.
A questo proposito si vede dai risultati dei test iniziali su MKL come l'ottimo venga già raggiunto con fusioni di 2 feature, nello specifico $SP_{RGBSIFT}$ e $GIST$, come a voler dimostrare che fra tutte le alternative derivanti in qualche modo da un insieme di feature puntuali quella che fornisce maggiori informazioni è la piramide spaziale calcolata a partire da informazioni che

includano dati sul colore; GIST d'altra parte fornisce informazioni a livello globale. Nel caso LDA invece le prestazioni migliori si hanno fondendo 3-4 feature; analizzando le configurazioni a 3, si nota che sia per i metodi *average* che per *product* vengono fuse 3 caratteristiche eterogenee, prese nella versione che fornisce il massimo contenuto informativo: $LDA_{RGBSIFT}$, $SP_{RGBSIFT}$ e $GIST$.
Da notare in conclusione dei test che i risultati di accuratezza massima di classificazione, 63.2%, ottenuti impiegando il modello a topic latenti LDA congiuntamente ad una piramide spaziale di feature puntuali e ad una descrizione a livello globale, portano ad un incremento intorno al 4% rispetto alle migliori prestazioni ottenute da Spatial Pyramid in configurazione unimodale. Questo risultato è in accordo con i valori riportati in [7], che mostrano come applicando un modello a topic latenti direttamente alla rappresentazione piramidale ed utilizzando per la classificazione un modello concettualmente simile a quello introdotto nel presente lavoro, le prestazioni di classificazione migliorano dal 64.6 ± 0.8 al 67.7 ± 1.5.

5.2 MICC-Flickr

I test eseguiti su Caltech-101 mostrano che i metodi che ottengono risultati migliori in termini di prestazioni di accuratezza sono quelli che si basano fortemente sull'analisi della componente spaziale della descrizione. Tale caratteristica trova giustificazione nell'analisi delle immagini che compongono il dataset: come è stato fatto notare all'inizio del capitolo, le immagini scelte sono state sottoposte a una serie di operazioni di preprocessing, ottenendo come risultato finale un insieme con scarsa variabilità intra-classe in termini di disposizione spaziale delle scene rappresentate. Gli oggetti sono in genere centrati ed occupano la maggior parte della scena; inoltre alcune immagini presentano degli angoli completamente neri, derivanti da rotazioni artificiali introdotte in modo che gli elementi al loro interno avessero orientazioni concordi. Come fatto notare in [37], queste caratteristiche possono portare a valori di prestazioni elevate probabilmente fuorvianti.
Nel primo capitolo si è d'altra parte mostrato l'interesse sempre crescente

che viene rivolto verso i dati multimediali pubblicati su Internet, sia per la grande quantità di informazione liberamente accessibile che per i metadati contenuti nelle descrizioni testuali sempre più spesso associate ad essa. Un dataset creato a partire da questi dati consente di utilizzare immagini comuni, realizzate spontaneamente dagli utenti ed in più di realizzare metodi di apprendimento multimodali che possono trarre beneficio dalle informazioni derivanti dall'analisi dei tag. Soluzioni come MIR-Flickr sono di sicuro interesse in questo senso, ma non consentono un confronto "diretto" con Caltech-101 viste le dissimilarità tra i due insiemi di classi.

Con queste premesse è stato realizzato il dataset denominato "MICC-Flick", costituito dalle stesse 101 classi di Caltech-101, composte però da immagini ottenute tramite ricerche su Flickr. Per ottenere un numero sufficiente di immagini per ogni classe è stato impiegato un software che consentisse di eseguire delle ricerche in base ai tag; in questo caso come parole chiave sono stati utilizzati i nomi stessi delle classi, tradotti in inglese, italiano, spagnolo, francese, e se necessario declinati da singolare a plurale. Ad esempio per la classe *lobster* sono state cercate immagini che fossero etichettate con *almeno uno* dei termini seguenti: *lobster*, *lobsters*, *langosta*, *langostas*, *aragosta*, *aragoste*, *homard*, *homards*. Le immagini prese in considerazione sono state quelle caricate negli ultimi 10 anni, vista la difficoltà nell'ottenere un numero sufficiente di immagini per alcune classi come *tick*, *wrench* o *brain*.
Tutte le immagini ottenute sono state esaminate dagli autori e dai correlatori del presente lavoro, scegliendo quelle da includere nell'insieme finale in modo che tutte le classi avessero un numero abbastanza simile di rappresentanti. Visto che i test compiuti su Caltech-101 prevedono di scegliere 30 rappresentati per classe per il train e fino a 50 per classe per il test, si è cercato di costruire classi di almeno 80 elementi. In questo modo si è oltretutto ottenuto un dataset più omogeneo rispetto al Caltech, che presenta forti disparità sul numero di elementi per classe; si veda a proposito la figura 5.7, in cui viene presentato un confronto tra le composizioni dei due dataset. Le immagini sono state inserite nel dataset così come sono state caricate dagli utenti, mantenendo risoluzione ed orientazione originali.

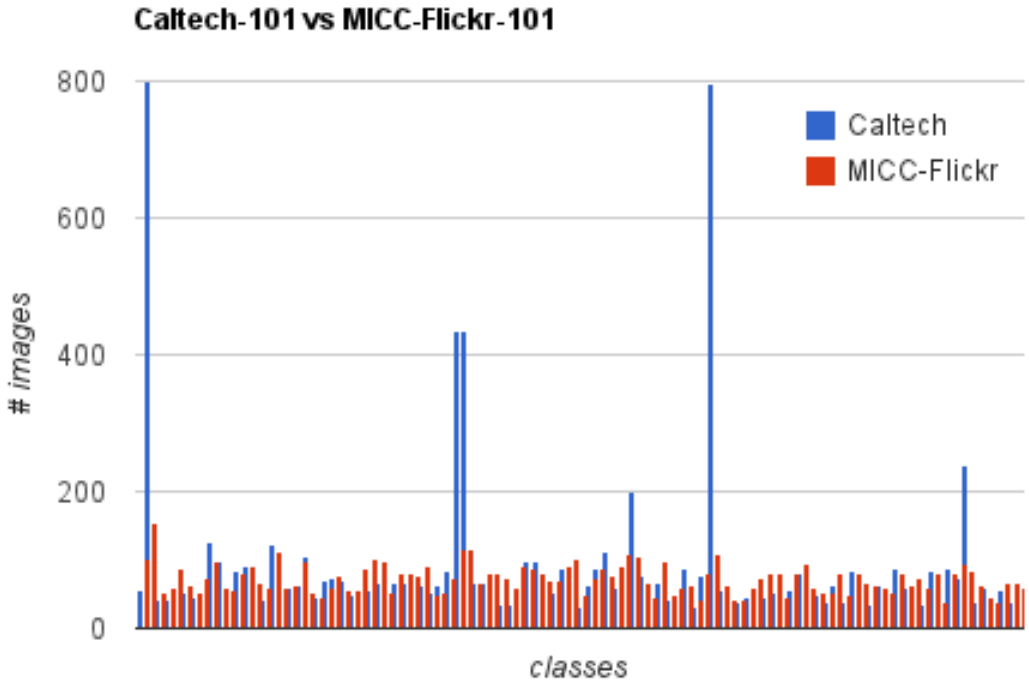


Figura 5.7. Confronto tra il numero di elementi per classe su Caltech-101 e MICC-Flickr-101

La principale difficoltà nel realizzare questo dataset è stata quella di mantenere un giusto equilibrio tra "realismo", "quotidianità" delle immagini e l'ovvia necessità di selezionare rappresentazioni sufficientemente accurate degli elementi delle diverse classi. L'uso di Flickr come sorgente ha infatti portato con sé uno svantaggio derivante dalla natura stessa del portale: la maggior parte degli utenti carica foto prevalentemente a carattere artistico, quindi può risultare difficile trovare rappresentazioni adeguate di oggetti come chiavi inglesi, cervelli umani e zecche. Altro criterio di cui si è tenuto conto è stato l'eventuale vincolo di ricercare immagini che mostrassero istanze singole, bene in evidenza degli oggetti desiderati, oppure se accettare anche rappresentazioni parziali o multiple degli stessi. Visti gli elementi ottenuti, è sembrato più interessante cercare di allentare i vincoli sulla rappresentazione, in modo da poter ottenere un dataset il più possibile realistico.

5.2.1 Metodi unimodali

I primi test eseguiti sul nuovo dataset sono stati compiuti con un modello BoW realizzato a partire da feature SIFT dense a singola scala, utilizzando come parametri iniziali quelli stabiliti per Caltech-101. I test al variare del numero di parole visuali costituenti il vocabolario sono riportati in figura 5.8. Stabilito in $W = 3000$ l'ottimo per questo nuovo insieme, sono stati eseguiti test con questi parametri sia per SIFT multiscala che per RGBSIFT. In accordo con i dati raccolti con Caltech-101 sui modelli a topic latenti, che mostravano l'ottimo in una riduzione di dimensionalità di un fattore 7.5, sono stati eseguiti test con entrambi i descrittori multiscala per $Z = 500$. I risultati ottenuti sono riportati nelle tabelle 5.6 e 5.7.

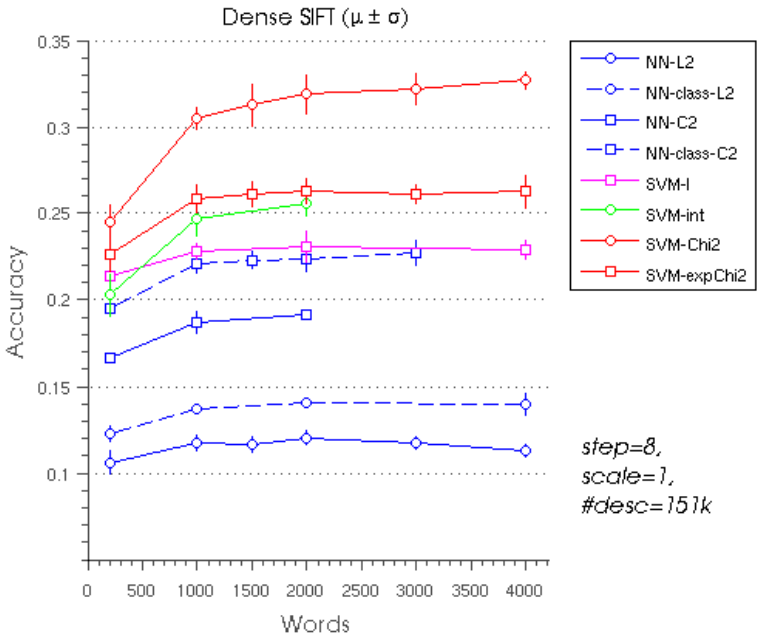


Figura 5.8. Prestazioni della pipeline basata su SIFT a singola scala

Estrazione di feature testuali

Utilizzando il software java LingPipe[1] per l'analisi di documenti testuali sono state realizzate delle rappresentazioni a bag of words e a topic latenti degli elementi del dataset: la componente testuale associata a ciascuna immagine è stata ricavata dall'unione dei termini contenuti nel titolo dell'immagine e dal set di tag ad essa associati. Il software offre in modo nativo degli strumenti per lo stemming e la rimozione delle stopwords per la lingua inglese; a questi sono state aggiunte informazioni sulle stopwords per le lingue italiana, spagnola e francese, in modo da permettere al software di interpretare con maggiore correttezza i dati forniti.
Per seguire un metodo di lavoro in linea con quello adottato nel caso di feature visuali, il vocabolario costruito per ogni split è stato composto considerando solo i termini presenti nei documenti di training: in pratica, fissata una soglia minima di occorrenze per poter considerare un *token* come un termine del dizionario, il contatore delle occorrenze viene incrementato solo nel caso in cui il termine compaia in un documento di addestramento. Tenendo conto di questo si può osservare come con gli stessi parametri un dizionario costruito su tutti i documenti del dataset arrivi alla dimensione di 1300, mentre utilizzando il metodo appena descritto la dimensione, in ogni caso variabile a seconda dello split, si attesti comunque intorno a 500 elementi. Si noti come la creazione del vocabolario per documenti testuali non sia vincolato, come nel caso del bag of words visuale, al parametro k del k-means che ne fissa la dimensione, non essendo necessaria per i tag alcuna operazione di clustering. Un modo per agire indirettamente sulla dimensione del vocabolario consiste invece nel variare la soglia di frequenza oltre la quale un *token* entra a far parte del dizionario: i due codebook di 1300 e 500 elementi presi ad esempio, costruiti rispettivamente sull'intero dataset il primo e sul training set il secondo, si ottengono impostando la frequenza minima del *token* ad un valore pari a 10.

Dall'analisi delle informazioni testuali per tutte le immagini del dataset

[1] `http://alias-i.com/lingpipe/`

sono stati realizzati sia un modello BoW che un LDA, così da ottenere due ulteriori rappresentazioni per ciascuna immagine.

Analisi di feature visuali

Le tabelle 5.6 e 5.7 mostrano i risultati di classificazione su MICC-Flickr-101 ottenuti dall'analisi di feature visuali.
Come anticipato nel precedente paragrafo, i vocabolari visuali utilizzati per le pipeline BoW sono composti da 3000 termini, per LDA è stato ricavato da questi vocabolari un insieme di 500 topic. Per le pipeline Spatial Pyramid è stato utilizzato un vocabolario di 400 termini, che su 3 livelli di piramide dà origine ad un descrittore di dimensione 8400; per la modalità GIST i test sono stati eseguiti con gli stessi parametri usati per il Caltech-101.

	L2	**class-L2**	χ^2	**class-**χ^2
$\mathbf{BoW_{SIFT}}$	14.1 ± 0.5	16.7 ± 0.2	-	-
$\mathbf{BoW_{RGBSIFT}}$	15.1 ± 0.9	$\pm$	-	-
$\mathbf{LDA_{SIFT}}$	11.8 ± 0.6	14.3 ± 0.4	18.5 ± 0.5	22.6 ± 0.7
$\mathbf{LDA_{RGBSIFT}}$	13.4 ± 0.6	–	–	–
$\mathbf{SP_{SIFT}}$	13.8 ± 0.1	–	–	–
GIST	16.0 ± 0.4	18.0 ± 1.0	-	-

Tabella 5.6. MICC-Flickr: risultati dell'analisi di feature visuali con metodi NN

Facendo un primo confronto tra i risultati esposti nelle tabelle e quelli visti per gli esperimenti sul Caltech-101 si nota come le pipeline unimodali offrano prestazioni peggiori per un dataset realistico come il MICC-Flickr-101, con una perdita in termini di accuratezza che supera il 13-14% per tutte le modalità, con picchi del 25% di riduzione per le modalità Spatial Pyramid e del 20% per GIST .
La tabella 5.8 mostra invece l'accuratezza media di classificazione delle classi per le quali si riscontra una notevole differenza di prestazioni passando dal dataset Caltech-101, valutato su una pipeline BoW SIFT con vocabolario

	Lin.	**Int.**	χ^2	$exp(\chi^2)$
$\mathbf{BoW_{SIFT\ 3000}}$	22.3 ± 0.4	30.6 ± 0.9	$\mathbf{32.1 \pm 1.0}$	31.2 ± 0.9
$\mathbf{BoW_{RGBSIFT\ 3000}}$	24.9 ± 0.7	33.0 ± 1.2	$\mathbf{34.4 \pm 0.6}$	–
$\mathbf{LDA_{SIFT\ 500}}$	21.9 ± 0.3	28.7 ± 0.9	28.9 ± 0.5	$\mathbf{29.3 \pm 0.9}$
$\mathbf{LDA_{RGBSIFT\ 500}}$	$\pm$	$\pm$	$\pm$	$\mathbf{32.5 \pm 0.6}$
$\mathbf{SP_{SIFT}}$	–	33.8 ± 0.7	$\mathbf{34.6 \pm 0.3}$	–
GIST	24.4 ± 0.1	$\mathbf{26.3 \pm 0.7}$	26.0 ± 0.4	26.1 ± 1

Tabella 5.7. MICC-Flickr: risultati dell'analisi di feature visuali con metodi SVM

di 1000 parole, al MICC-Flickr-101, valutato sulla stessa pipeline ma con vocabolario composto da 3000 termini. Nelle figure 5.9 e 5.10 viene riportato il confronto delle prestazioni di classificazione per categoria ottenuto per tutte le classi.

	Caltech-101	**MICC-Flickr-101**
accordion	90.667	30.67
mandolin	15.39	01.67
menorah	39.33	09.80
soccerball	50.00	02.22

Tabella 5.8. Confronto delle prestazioni per alcune categorie di Caltech-101 e MICC-Flickr-101

Il peggioramento delle prestazioni può essere motivato essenzialmente con alcune considerazioni sulla diversa rappresentazione degli oggetti nei due dataset: osservando le figure 5.11 e 5.12, in cui sono raccolte alcune immagini delle classi *soccerball* e *chair* nei due dataset, si possono notare alcune caratteristiche sostanziali che differenziano le due collezioni:

- **Contesto di rappresentazione**. Le immagini in Caltech-101 sono composte dall'oggetto rappresentativo della categoria disposto su uno sfondo generalmente monocromatico in cui non sono presenti altri og-

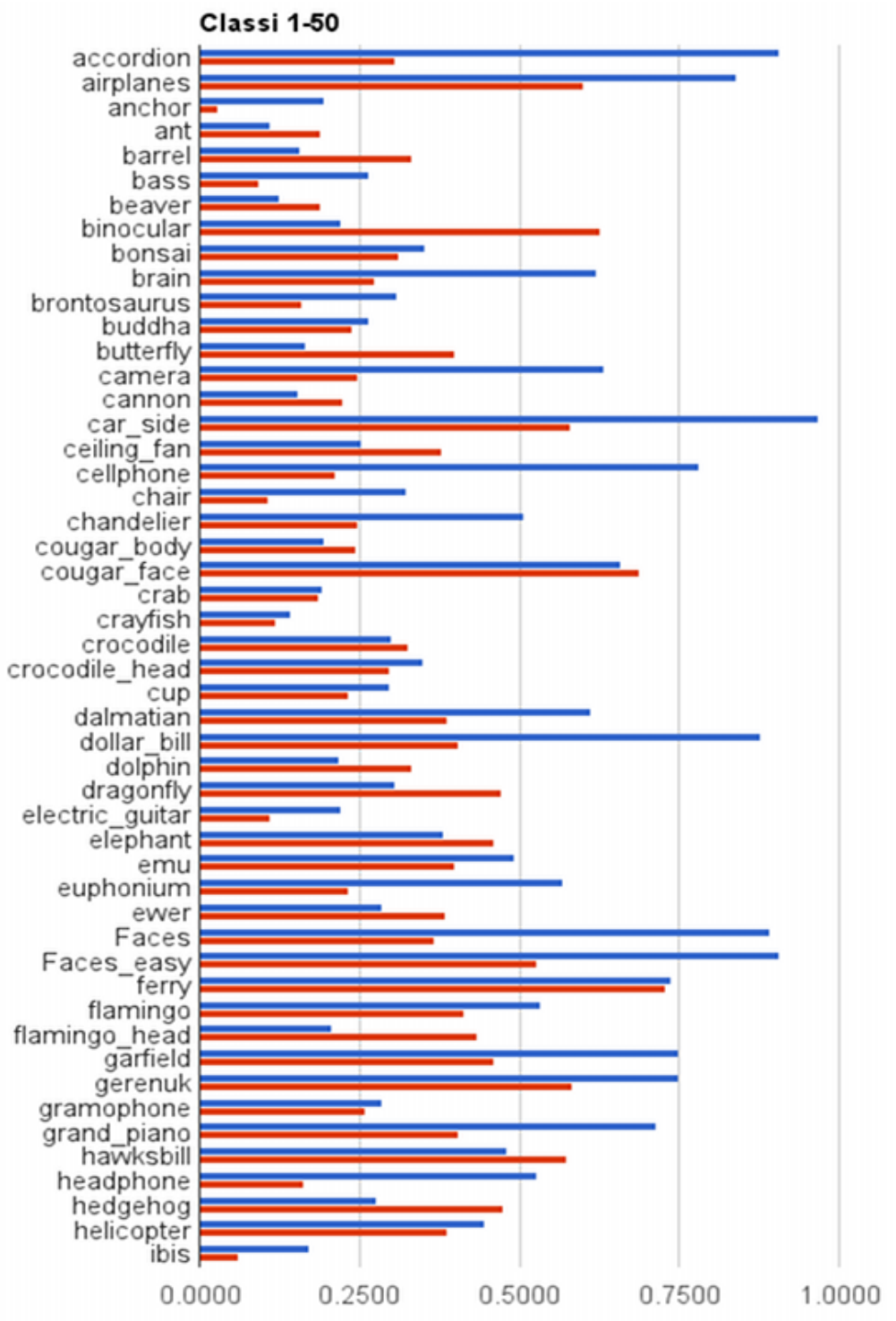


Figura 5.9. Confronto delle prestazioni per classe per Caltech-101 (blu) e MICC-Flickr-101 (rosso) per le prime 50 classi ($W_{Caltech} = 1000$, $W_{MICC-Flickr} = 3000$)

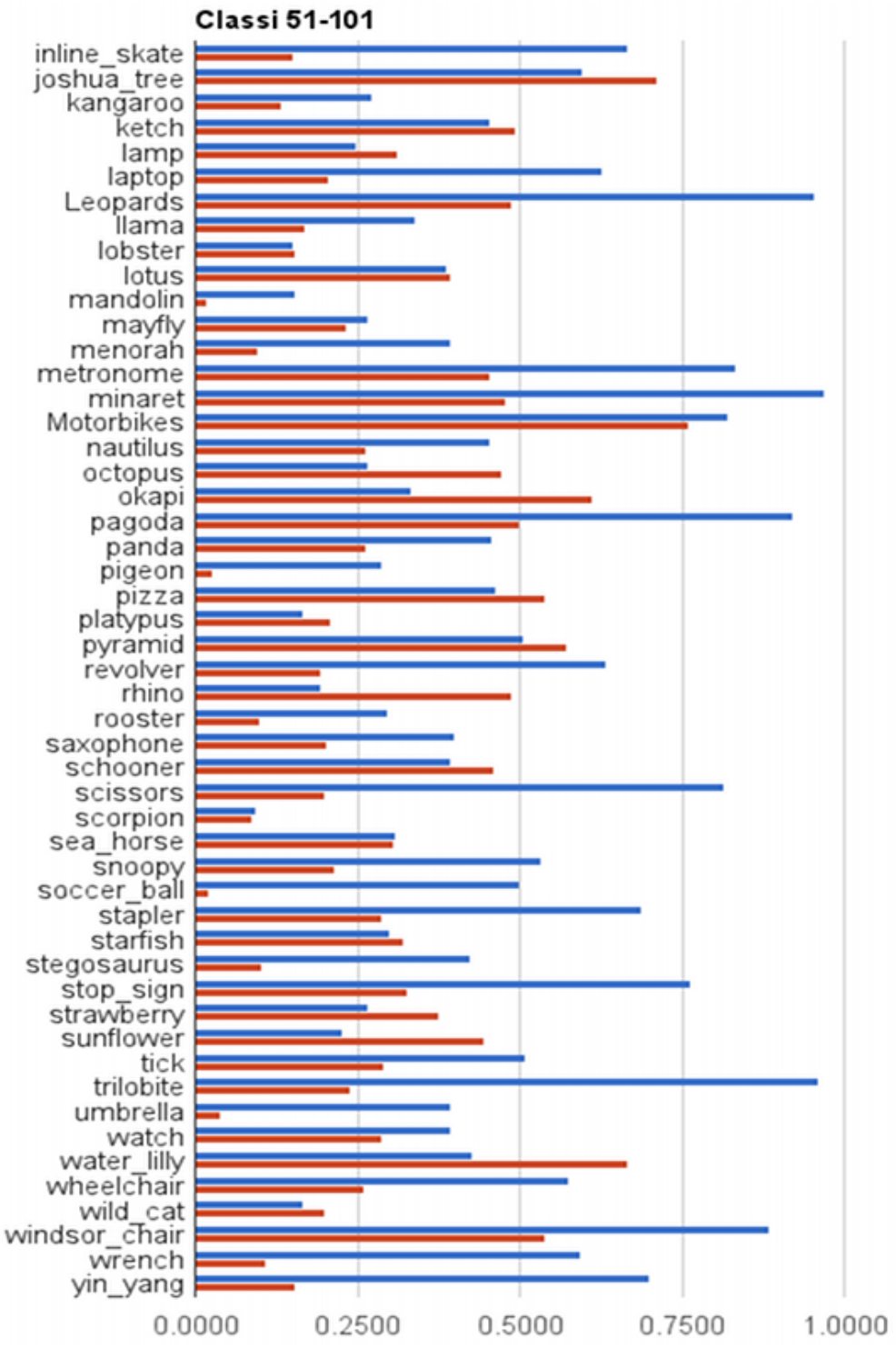


Figura 5.10. Confronto delle prestazioni per classe per Caltech-101 (blu) e MICC-Flickr-101 (rosso) per le classi da 51 a 101 ($W_{Caltech} = 1000$, $W_{MICC-Flickr} = 3000$)

getti; il contesto in cui sono ripresi gli oggetti di MICC-Flickr-101 è invece molto variabile ed in molti casi l'immagine include anche altri oggetti oltre a quello appartenente alla categoria di riferimento.

- **Variabilità intra-classe**. Gli oggetti in Caltech-101 sono tutti centrati ed orientati nella stessa direzione, inoltre gli oggetti appartenenti alla stessa classe hanno tutti lo stesso aspetto, come dimostrano le immagini del pallone da calcio visibili nella figura 5.11. Il dataset MICC-Flickr-101, al contrario, è stato costruito mantenendo una grande variabilità intra-classe per gli oggetti appartenenti alla stessa categoria, in modo da simulare il più possibile la variabilità che gli oggetti hanno nel mondo reale.

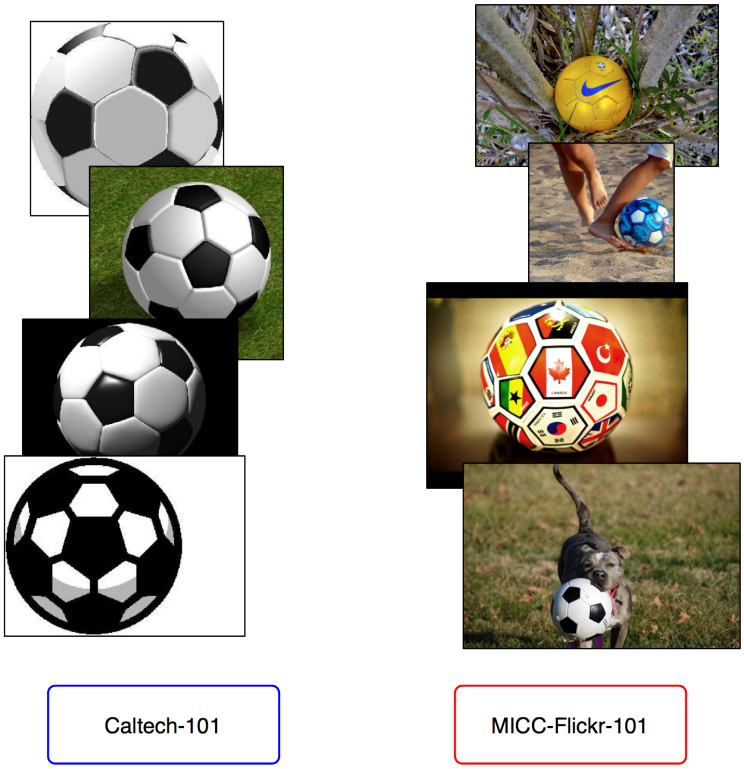


Figura 5.11. Rappresentazione della classe *soccerball* in Caltech-101 ed in MICC-Flickr-101

Figura 5.12. Rappresentazione della classe *chair* in Caltech-101 ed in MICC-Flickr-101

Con le considerazioni appena fatte appare dunque ampiamente giustificabile il calo delle prestazioni di accuratezza riscontrato con il MICC-Flick-101, per il quale si ottiene comunque un valore massimo del 34% per pipeline unimodali basate su feature locali.

Analisi di feature testuali

Le tabelle 5.9 e 5.10 mostrano i risultati ottenuti classificando le immagini di MICC-Flickr-101 a partire dalle feature testuali.

I risultati ottenuti nei primi test di classificazione riportano valori elevati di accuratezza, dovuti principalmente a due motivi:

1. **Numero ridotto di feature per immagine**: le immagini di MICC-

	L2	**class-L2**	χ^2	**class-**χ^2
$\mathbf{BoW_{tags\ 500}}$	68.3 ± 0.4	74.4 ± 0.4	69.4 ± 0.4	70.0 ± 0.5
$\mathbf{LDA_{tags\ 50}}$	31.4 ± 0.8	-	-	-
$\mathbf{LDA_{tags\ 100}}$	49.4 ± 0.5	57.6 ± 0.8	47.3 ± 0.2	58 ± 1.2

Tabella 5.9. MICC-Flickr: risultati dell'analisi di feature testuali con metodi NN

Flickr-101 contano in media un insieme di 4-5 termini, ottenuti dall'unione delle parole che compaiono nel titolo e dai tag aggiunti dall'utente che ha pubblicato il documento: considerando un vocabolario di termini di almeno 500 parole, si ottiene per ciascuna immagine un vettore descrittore di tipo BoW estremamente sparso.

2. **Potere discriminativo di alcuni termini**: il dataset è composto da immagini ottenute ricercando su Flickr i nomi delle classi presenti in Caltech-101. Il software utilizzato per questa operazione restituisce le immagini che contengono nei metadati, e quindi anche nel titolo o nell'insieme dei tag, le chiavi della ricerca: questo significa che l'insieme delle feature testuali di ciascuna immagine contiene anche il nome della classe di appartenenza, che potrebbe pertanto risultare estremamente discriminativo per le operazioni di classificazione.

	Lin.	**Int.**	χ^2	$exp(\chi^2)$
$\mathbf{BoW_{tags\ 500}}$	71.2 ± 1.1	77.8 ± 0.3	76.6 ± 0.7	79.1 ± 1.0
$\mathbf{LDA_{tags\ 50}}$	41.2 ± 0.3	41.7 ± 0.9	40.8 ± 0.8	39.6 ± 0.6
$\mathbf{LDA_{tags\ 100}}$	59.1 ± 1.8	59.4 ± 1	58.9 ± 1.1	59.6 ± 2.2

Tabella 5.10. MICC-Flickr: risultati dell'analisi di feature testuali con metodi SVM

Le caratteristiche delle feature testuali che accompagnano le immagini sono una logica conseguenza della metodologia seguita per la realizzazione del dataset; se la presenza di pochi termini per immagine è un compromesso

inevitabile se si vuole utilizzare informazione proveniente da un social network come Flickr, il peso discriminativo delle chiavi di ricerca esposto nel punto 2) può essere ridotto a patto di modificare il set di tag associati alle immagini.
Per questo motivo una seconda serie di esperimenti è stata effettuata aggiungendo all'insieme delle stopwords i nomi delle classi, ovvero le chiavi di ricerca utilizzate per comporre il dataset: pur essendo questa una modifica al set di feature inizialmente presente nel documento e quindi un'alterazione del contenuto informativo originario, è sembrato interessante misurare le prestazioni delle pipeline unimodali rinunciando al potere descrittivo dei termini in questione. Le tabelle 5.11 e 5.12 mostrano i risultati ottenuti classificando MICC-Flickr con pipeline BoW ed LDA basate su vocabolari privi dei nomi delle classi, a cui si fa riferimento con il pedice $_{kr}$. Gli esperimenti sul nuovo vocabolario sono stati eseguiti abbassando la soglia della frequenza minima dei *token*, ottenendo così un insieme di circa 1000 termini; le prestazioni del vocabolario così ottenuto, rispetto ad un analogo dizionario di 500 termini privo dei nomi delle classi, sono superiori per quasi tutti i metodi di classificazione di circa il 5%.

	L2	**class-L2**	χ^2	**class-**χ^2
$\mathbf{BoW_{tags_{kr}\ 1000}}$	37.5 ± 0.9	41.6 ± 1.2	38.8 ± 0.4	39.1 ± 0.6
$\mathbf{LDA_{tags_{kr}\ 100}}$	26.1 ± 1.3	34.1 ± 0.9	24.3 ± 1.0	33.7 ± 0.7

Tabella 5.11. MICC-Flickr: risultati dell'analisi di feature testuali con metodi NN su vocabolari privi delle chiavi di ricerca

Come era prevedibile sottraendo al dizionario alcuni termini estremamente discriminativi per la descrizione dei documenti le prestazioni non possono che peggiorare, anche se si riesce a raggiungere in ogni caso un'accuratezza di classificazione del 50.9% per la pipeline basata su BoW e del 37.2% per la pipeline basata su LDA con 100 topic.

	Lin.	**Int.**	χ^2	$exp(\chi^2)$
$\mathbf{BoW_{tags_{kr}\ 1000}}$	42.9 ± 0.5	47.8 ± 1.4	47.7 ± 1.4	$\mathbf{50.9 \pm 1.6}$
$\mathbf{LDA_{tags_{kr}\ 100}}$	35.8 ± 1.5	$\mathbf{37.2 \pm 1.0}$	36.4 ± 1.4	36.4 ± 1.3

Tabella 5.12. MICC-Flickr: risultati dell'analisi di feature testuali con metodi SVM su vocabolari privi delle chiavi di ricerca

5.2.2 MKL

Le pipeline unimodali utilizzate per i test appena descritti sono state combinate in configurazioni di Multiple Kernel Learning secondo le metodologie già viste per il Caltech-101. La tabella 5.13 mostra i risultati ottenuti fondendo modalità basate sul modello BoW sia per feature visuali che per i tag, mentre la tabella 5.14 mostra le prestazioni ottenute introducendo i modelli

	Average		**Product**	
#	value	feats	value	feats
2	58.8 ± 1.1	SP_{SIFT} *int* BoW_{tags} $exp(\chi^2)$	59.3 ± 1.1	SP_{SIFT} *int* BoW_{tags} $exp(\chi^2)$
3	58.7 ± 1.5	$BoW_{RGBSIFT}$ χ^2 $GIST$ $exp(\chi^2)$ BoW_{tags} *int*	$\mathbf{61.7 \pm 0.6}$	**SP_{SIFT} int** **GIST int** **BoW_{tags} exp(χ^2)**
4	$\mathbf{60.0 \pm 1.5}$	**BoW_{SIFT} exp(χ^2)** **SP_{SIFT} int** **GIST exp(χ^2)** **BoW_{tags} int**	59.8 ± 0.3	BoW_{SIFT} $exp(\chi^2)$ SP_{SIFT} *int* $GIST$ *int* BoW_{tags} $exp(\chi^2)$
5	59.8 ± 1.0	$BoW_{RGBSIFT}$ χ^2 BoW_{SIFT} $exp(\chi^2)$ SP_{SIFT} *int* $GIST$ $exp(\chi^2)$ BoW_{tags} *int*	45.1 ± 1.0	$BoW_{RGBSIFT}$ χ^2 BoW_{SIFT} $exp(\chi^2)$ SP_{SIFT} *int* $GIST$ $exp(\chi^2)$ BoW_{tags} *int*

Tabella 5.13. Risultati MKL per fusioni a 2,3,4 e 5 feature utilizzando modelli bag of words

a topic latenti.

	Average		**Product**	
#	value	feats	value	feats
2	53.2 ± 0.7	SP_{SIFT} *int* LDA_{tags} *int*	50.0 ± 1.1	$LDA_{RGBSIFT}$ $exp(\chi^2)$ LDA_{tags} $exp(\chi^2)$
3	54.7 ± 1.3	$LDA_{RGBSIFT}$ $exp(\chi^2)$ $GIST$ $exp(\chi^2)$ LDA_{tags} χ^2	53.9 ± 0.8	$LDA_{RGBSIFT}$ $exp(\chi^2)$ $GIST$ $exp(\chi^2)$ LDA_{tags} $exp(\chi^2)$
4	55.9 ± 1.4	LDA_{SIFT} *int* $LDA_{RGBSIFT}$ $exp(\chi^2)$ $GIST$ $exp(\chi^2)$ LDA_{tags} χ^2	$\mathbf{55.4 \pm 1.0}$	$\mathbf{LDA_{RGBSIFT}}$ $\mathbf{exp}(\chi^2)$ $\mathbf{SP_{SIFT}}$ **int** **GIST** $\mathbf{exp}(\chi^2)$ $\mathbf{LDA_{tags}}$ $\mathbf{exp}(\chi^2)$
5	$\mathbf{58.8 \pm 0.9}$	$\mathbf{LDA_{SIFT}}$ **int** $\mathbf{LDA_{RGBSIFT}}$ $\mathbf{exp}(\chi^2)$ $\mathbf{SP_{SIFT}}$ **int** **GIST** $\mathbf{exp}(\chi^2)$ $\mathbf{LDA_{tags}}$ χ^2	55.3 ± 0.9	$LDA_{RGBSIFT}$ $exp(\chi^2)$ LDA_{SIFT} $exp(\chi^2)$ SP_{SIFT} *int* $GIST$ $exp(\chi^2)$ LDA_{tags} $exp(\chi^2)$

Tabella 5.14. Risultati MKL per fusioni a 2,3,4 e 5 feature utilizzando modelli a topic latenti

La tabella 5.15 mostra i risultati delle pipeline che realizzano le migliori prestazioni, ovvero le combinazioni evidenziate in grassetto nelle tabelle precedenti, private del contributo dato dall'analisi delle feature testuali. Nella tabella 5.16 vengono riportati infine i risultati delle migliori fusioni in assoluto; da notare che sia in caso average che per product la massimo incremento di accuratezza si ottiene utilizzando LDA per le feature visuali, BoW per quelle derivanti dall'analisi dei tag.

5.2.3 Valutazione delle prestazioni

Le prestazioni delle pipeline unimodali valutate su MICC-Flickr-101 differiscono a seconda del tipo di informazione utilizzata per l'addestramento.

Average		**Product**	
value	feats	value	feats
38.0 ± 1.5	**BoW$_{\text{SIFT}}$ exp(χ^2)** **SP$_{\text{SIFT}}$ int** **GIST exp(χ^2)**	**37.4 ± 0.7**	**SP$_{\text{SIFT}}$ int** **GIST int**
38.4 ± 1.7	**LDA$_{\text{SIFT}}$ int** **SP$_{\text{SIFT}}$ int** **GIST exp(χ^2)**	**38.3 ± 1.1**	**LDA$_{\text{SIFT}}$ exp(χ^2)** **SP$_{\text{SIFT}}$ int** **GIST exp(χ^2)**

Tabella 5.15. Risultati MKL per feature di tipo visuale sul dataset MICC-Flickr

Average		**Product**	
value	feats	value	feats
62.0 ± 1.2	**LDA$_{\text{RGBSIFT}}$ exp(χ^2)** **LDA$_{\text{SIFT}}$ exp(χ^2)** **SP$_{\text{SIFT}}$ int** **GIST exp(χ^2)** **BoW$_{\text{tags}}$ int**	**62.3 ± 0.3**	**LDA$_{\text{RGBSIFT}}$ exp(χ^2)** **GIST int** **BoW$_{\text{tags}}$ expChi2**

Tabella 5.16. Migliori risultati MKL per fusioni a 2,3,4 o 5 feature per il dataset MICC-Flickr

Nel caso di feature visuali è stato precedentemente spiegato come la perdita in termini di accuratezza rispetto ai valori registrati sul Caltech-101 sia giustificabile con un'analisi del contenuto delle immagini del nuovo dataset: gli oggetti sono qui rappresentati in scene generalmente più complesse, sia a livello di composizione semantica, data la presenza di altri concetti oltre a quello rappresentativo della classe, sia in termini di variabilità nella rappresentazione visiva della categoria di riferimento.
Da queste considerazioni segue la necessità di utilizzare dei vocabolari di dimensioni maggiori rispetto a quelli scelti per Caltech-101 per codificare un insieme di informazioni visuali più ampio ed eterogeneo; allo stesso tempo la maggiore quantità di concetti visuali rende più difficile la classificazione

di immagini che rappresentano contenuti più complessi di quelli presenti nel Caltech.
Alla luce di tutte queste considerazioni le prestazioni delle pipeline unimodali basate sul contenuto visuale sono da considerarsi in ogni caso buone: in particolare i valori migliori si ottengono con modelli basati su feature locali, come BoW ed LDA, per cui si raggiungono rispettivamente accuratezze del 32-34% e del 29-32%. Anche questo set di esperimenti conferma come i modelli a topic latenti, a fronte di una perdita media di accuratezza del 2-3% rispetto al metodo BoW con cui di volta in volta condividono il vocabolario visuale, consentano di ottenere una riduzione dimensionale dello spazio di descrizione di un fattore 6.

Un discorso diverso in questo caso va fatto per GIST e Spatial Pyramid per le cui pipeline si riscontra, a differenza di quanto visto per il dataset precedente, un incremento minimo o addirittura una diminuzione delle prestazioni rispetto al modello classico di bag of words.
Per quanto riguarda il primo metodo bisogna precisare però che il descrittore usato per MICC-Flickr-101 è del tutto identico a quello usato per Caltech-101: questa scelta, legata principalmente ai tempi ristretti in cui è stata condotta questa ultima sessione di test, lascia però intendere che con un nuovo descrittore GIST, dimensionato opportunamente su questo dataset con l'aggiunta di altre orientazioni e scale alle quali applicare i filtri, le prestazioni potrebbero aumentare in maniera considerevole.
La riduzione di prestazioni per il modello Spatial Pyramid applicato al nuovo dataset era in una qualche misura prevedibile, data la mancanza di una rigida disposizione spaziale caratteristica degli oggetti rappresentati in Caltech-101. L'assenza di un sostanziale vantaggio nell'utilizzo della piramide spaziale può essere visto come conferma del maggior grado di generalità del dataset MICC-Flickr-101.

Le prestazioni delle pipeline basate su feature testuali hanno raggiunto valori differenti a seconda del criterio utilizzato per la composizione del vocabolario. La scelta di escludere dal vocabolario i nomi delle classi, evitando

così che alcuni termini chiave risultassero estremamente discriminativi a causa anche della ridotta quantità di feature testuali per documento, ha portato ad un drastico calo dei valori di accuratezza ristabilendo le prestazioni migliori su valori del 50.9% e del 37.2% per modelli BoW ed LDA. Nel caso di analisi di feature testuali, inoltre, il gap tra i metodi NN e quelli SVM si riduce utilizzando modelli a topic latenti.

L'analisi dei risultati ottenuti dal Multiple Kernel Learning evidenzia come, rispetto alle pipeline unimodali, questo metodo porti ad un incremento di accuratezza del 11% su modelli bag of words e quasi del 22% utilizzando i topic latenti. Nello specifico si passa nel primo caso da un risultato del 34.6 % per pipeline costruite su feature visuali e del 50.9% per pipeline definite sul testo ad un valore massimo del 61.7%; nel secondo caso invece da accuratezze massime del 32.5% e del 37.2% si ottiene un risultato del 58.8%. Da notare che tutte le combinazioni migliori comprendono per la parte visuale un insieme di pipeline eterogenee con l'aggiunta della pipeline costruita sull'analisi testuale. In particolare la tabella 5.15 mostra come il miglioramento che si avrebbe combinando soltanto feature visuali, pari a circa il 4%, viene incrementato dall'introduzione dell'analisi dei tag di un 20% per modelli che usano solo LDA e di un 23% per il solo BoW.
Per concludere si osserva come i risultati migliori in termini assoluti siano stati ottenuti fondendo un modello bag of words costruito a partire dai tag con delle pipeline visuali che comprendono un modello LDA; i corrispondenti valori di prestazioni sono riportati in tabella 5.16.

Conclusioni

In questo lavoro sono stati introdotti alcuni approcci differenti alla descrizione delle immagini ed è stato fornito, per ciascuno di essi, un modello progettuale che ne prevede l'uso all'interno di una pipeline di classificazione. Alcune delle metodologie viste partono da presupposti simili, come nel caso di metodi basati su feature locali, ed utilizzano un approccio condiviso come il bag of words. Tale metodo viene spesso esteso per definire un livello di descrizione più dettagliato, come succede per lo Spatial Pyramid Matching, o più raffinato dal punto di vista semantico, come accade nei modelli a topic latenti. In particolare l'uso di un modello come LDA offre alla descrizione di un documento un livello di astrazione più alto, laddove al semplice conteggio statistico delle relazioni che intercorrono tra punti salienti e parole visuali si sostituisce un modello di inferenza dei concetti che hanno generato il documento stesso.

Queste considerazioni di carattere teorico trovano riscontro nell'insieme degli esperimenti effettuati: la descrizione mediante topic latenti, infatti, risulta essere una più che valida alternativa al modello classico di bag of words. Gli esperimenti dimostrano oltretutto che l'uso di un modello generativo come LDA è in grado di ottenere prestazioni addirittura migliori del BoW nel caso in cui si decida di rinunciare ad uno dei vantaggi di questo modello, ovvero la riduzione dimensionale, confrontando quindi in entrambi i casi le prestazioni ottenute con descrittori di pari dimensione.

Anche l'idea di vincolare l'analisi di un documento ad una struttura geometrica che stabilisce l'ordine ed il peso con cui i punti salienti debbano contribuire al descrittore, come succede per lo Spatial Pyramid Matching, sembra apportare un miglioramento rispetto alla rappresentazione *orderless*

del metodo di base: resta chiaro però che il vantaggio di una struttura preimpostata come la piramide spaziale è tanto maggiore quanto più risultano localizzabili e "privi di rumore" i concetti contenuti nelle immagini, cosa che si verifica pienamente nel Caltech-101.

I risultati degli esperimenti hanno anche confermato come sia possibile trarre vantaggio da più rappresentazioni differenti attraverso un utilizzo congiunto delle pipeline che le realizzano: i valori di accuratezza raggiunti con i metodi di Multiple Kernel Learning mostrano, seppur nelle modalità *baseline*, che un processo complesso come la classificazione di una collezione di dati venga migliorato dall'integrazione di rappresentazioni tra di loro complementari.
Proprio l'uso di informazioni di natura diversa e, di conseguenza, rappresentazioni diverse di concetti analoghi, è stato uno dei motivi per la creazione del dataset MICC-Flickr-101. Come già detto questa nuova collezione è stata concepita con l'idea di clonare un dataset di riferimento come Caltech-101, utilizzando però dei documenti realistici e rappresentati attraverso descrizioni anche di tipo testuale.
Pur essendo l'interesse del lavoro principalmente orientato ad un'analisi delle feature visuali, si è potuto notare quanto lo studio delle informazioni testuali possa essere integrato alle metodologie applicate alle immagini e quanto l'utilizzo di pipeline multimodali costruite su questi aspetti complementari possa risultare utile in operazioni non solo di classificazione, ma anche di tag suggestion o di Information Retrieval.
In definitiva la fusione di kernel derivanti da apprendimento su feature diverse, visuali per Caltech-101 e visuali/testuali per MICC-Flickr-101, ha aumentato le prestazioni del sistema di classificazione proponendo, tra le configurazioni a più alto guadagno, quelle costruite combinando approcci estremamente eterogenei tra di loro.

Capitolo 6

Dettagli sull'implementazione

In questa appendice viene proposta una breve guida al software che è stato sviluppato sulla base dei modelli teorici e degli schemi progettuali illustrati nei precedenti capitoli.

6.1 Risorse

Tutti i test sono stati eseguiti in ambiente Linux, utilizzando script di Matlab per la costruzione e valutazione di modelli BoW, oltre a risorse software esterne per il calcolo di descrittori, modelli ad argomenti latenti e Spatial Pyramid. In una prima fase la quasi totalità degli esperimenti su Caltech ed alcuni degli esperimenti su MICC-Flickr sono stati condotti su una macchina con processore Intel Xeon quad-core 2.33 GHz e 11 GB di RAM. I test più complessi a livello di risorse computazionali necessarie sono stati condotti su un cluster recentemente installato presso i laboratori del MICC. Tale macchina è costituita da:

- 4 nodi *elaboratori*:
  - CPU: 4 Core (2 reali, 4 logici)
  - RAM: 8 GB
  - HDD: 320GB 7.2k rpm
- 1 nodo *master*

  - CPU: 4 Core (2 reali, 4 logici)
  - RAM: 16 GB
  - HDD: 320GB 10k rpm

- 1 dispositivo SAS
  - HDD: 4 TB (RAID0)
  - connessione in fibra ottica

Il SO installato sul cluster è Linux distribuzione Rocks Cluster, derivata da CentOS.

6.2 Implementazione

Per i test eseguiti durante la prima fase è stata usata la versione 2009a di Matlab; i test sul cluster sono stati invece condotti con la R2011a.
Nella versione di partenza del software erano inclusi uno script per il calcolo di K-means basato su [14] e la versione per Matlab di libsvm 2.86[1]. Tale versione consentiva l'esecuzione di una pipeline unimodale, dall'estrazione di feature SIFT a singola scala dense, per mezzo di un'implementazione di Lazebnik e Torralba, fino alla classificazione tramite NN o SVM, passando da creazione del codebook, quantizzazione e creazione dell'istogramma. Erano già presenti script per il calcolo delle distanze L2 e χ^2 tra coppie di vettori, oltre che per i kernel χ^2 ed intersection.
Durante la realizzazione del presente lavoro sono state aggiunte le funzionalità per il calcolo delle distanze NN-class e per il calcolo del kernel $exp(\chi^2)$, sempre nella forma di script Matlab integrati nella versione originale. Il codice è stato inoltre sottoposto ad attività di refactoring, per la creazione di metodi che potessero essere invocati separatamente. Ad oggi il software è in grado di:

- eseguire l'estrazione delle feature;

[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

- creare *split* del dataset da analizzare, ovvero scelte casuali di un numero fissato di immagini di test e di train per ogni classe; tali scelte possono essere mantenute fisse anche al variare dei descrittori utilizzati, in modo da confrontare correttamente le diverse prestazioni;
- scegliere un sottoinsieme dei descrittori delle immagini di addestramento per realizzare il clustering k-means; tali elementi vengono selezionati in modo casuale dall'insieme complessivo o specificando una quantità fissa per ogni immagine;
- creare un codebook a partire da una scelta di descrittori e salvarlo per usi futuri, oppure caricarne uno già computato in precedenza;
- calcolare le prestazioni in termini di accuratezza di classificazione per tutti i metodi NN ed SVM introdotti in precedenza, salvando inoltre al termine dell'esecuzione i kernel computati e le matrici di confusione.

Come parametro di valutazione delle prestazioni è stata usata l'accuratezza di classificazione per classe, ovvero la percentuale di immagini di test correttamente classificate per ogni classe. Per far si che tali valori fossero invarianti rispetto al numero di elementi per classe e per evitare che classi con molti esempi pesassero eccessivamente sul risultato finale, i valori di accuratezza sono stati normalizzati rispetto al numero di immagini che compongono ciascuna categoria.
Di seguito viene mostrato un elenco dei principali moduli che compongono il software, riportati secondo l'ordine di invocazione durante l'esecuzione delle pipeline:

- `run_bow_params`: file di configurazione che specifica quali azioni deve compiere il codice ed i parametri di configurazione per i vari metodi implementati; in caso si vogliano eseguire più esperimenti, variando i parametri di configurazione, le tipologie di descrittore o i metodi di descrizione da utilizzare, basta creare una versione personalizzata di questo file;
- `run_bow_engine`: script di esecuzione principale, fa da contenitore per tutti gli altri e si occupa di salvare i risultati;

- `extract_<feat>_features`: con `feat` che assume valori diversi a seconda del descrittore, si occupa dell'estrazione delle diverse tipologie di feature utilizzate;
- `split_create_random`: seleziona un insieme casuale di immagini di train e test per la costiuzione dei vari *split*; da notare che l'implementazione corrente consente di mantenere costante la scelta delle immagini, fissato lo split corrente, al variare dei descrittori: in questo modo i test sull'i-esimo split sono sempre eseguiti sull'i-esimo insieme di elementi;
- `split_dataset`: crea una serie di collegamenti ai file in cui sono salvati i descrittori per ogni split;
- `split_save`: usato per l'analisi di feature locali, salva un insieme casuale di descrittori presi dalle immagini di training, che verranno impiegati per la creazione del codebook;
- `bow_pipeline_codebook`: crea un descrittore a livello immagine per ogni elemento di train e di test: per BoW classico, carica o crea il codebook per ogni split e provvede a creare il vettore di frequenza di parole visuali per ogni immagine; per LDA realizza la descrizione come probabilità dei diversi topic del modello; per Spatial Pyramid Matching restituisce la corrispondente descrizione multilivello; per GIST calcola il descrittore globale;
- `bow_pipeline_NN`: test delle rappresentazioni delle immagini con le tecniche NN precedentemente introdotte;
- `bow_pipeline_SVM`: creazione delle matrici kernel e test delle rappresentazioni delle immagini con le tecniche SVM precedentemente introdotte;

L'esecuzione principale viene avviata utilizzando `run_bow_params`, che si occupa di specificare l'elenco dei parametri e poi invoca `run_bow_engine`.
A questi moduli sono da aggiungere gli script per la gestione del MKL: `run_mkl_bow` per impostare i parametri per bag of words; `run_mkl_topics` in caso si usi LDA; `run_mkl_enigne` gli script che consentono di eseguire le

fusioni vere e proprie. Una serie di script, identificati da prefisso `plot_`, sono stati inoltre realizzati per ottenere dei grafici riassuntivi sui risultati ottenuti con i vari metodi.

Nella seconda fase dei test, quella eseguita su cluster, sono state apportate alcune modifiche al codice per consentire di sfruttare le potenzialità della macchina eseguendo delle istruzioni in parallelo. Essenzialmente sono stati impiegati, in casi opportuni, dei costrutti `parfor` al posto dei classici `for`. I cicli del primo tipo, infatti, se utilizzati attivando l'ambiente `matlabpool`, consento l'esecuzione parallela delle iterazioni del ciclo. Per usufruire dell'architettura parallela esistono alcuni approcci diversi; quello usato nel presente lavoro consiste nell'eseguire blocchi di codice di questo tipo:

```
matlabpool open 8;
run_bow_params;
matlabpool close;
```

La prima istruzione riserva un totale di 8 core, se disponibili tra tutti quelli delle macchine elaboratrici, per l'esecuzione dello script caricato dalla seconda istruzione. L'ultima istruzione rilascia le risorse allocate.

6.2.1 ColorSIFT

Per il calcolo dei SIFT multiscala e delle relative versioni a colori si è fatto uso della versione 3.0 dell'implementazione di Koen van de Sande [39] del *colorDescriptor* introdotto in [40][2]. Per invocare lo script si usa la seguente sintassi:

```
colorDescriptor <image> --detector <detector>
                        --descriptor <descriptor>
                        --output <descriptorfile.txt>
```

Il campo `--detector` può assumere due valori diversi, associati ai simboli `harrislaplace` o `densesampling`, a seconda si preferisca utilizzare un cam-

[2] `http://koen.me/research/colordescriptors/`

pionamento di tipo sparso o denso. In caso si utilizzi il metodo denso è possibile specificare il passo e la scala a cui estrarre i descrittori:

```
--ds_spacing pixels [default: 6]
--ds_scales scale1+scale2+...
```

I possibili descrittori sono quelli già introdotti in precedenza, più dei semplici istogrammi di colore:

- SIFT (`sift`);
- HueSIFT (`huesift`);
- HSV-SIFT (`hsvsift`);
- OpponentSIFT (`opponentsift`);
- *rg*SIFT (`rgsift`);
- C-SIFT (`csift`);
- RGB-SIFT (`rgbsift`);
- transformed color SIFT (`transformedcolorsift`);

I file di output possono essere salvati in formato binario aggiungendo l'opzione `--outputformat binary`; in questo caso l'autore mette a disposizione uno script Matlab per leggere il file contenente i descrittori[3].

6.2.2 Spatial Pyramid Matching

Per i test sul modello spatial pyramid si è utilizzata l'implementazione di Tighe e Lazebnik del 2010[4]. La funzione principale per la costruzione della piramide spaziale è `BuildPyramid`. La versione originale per prima cosa estrae descrittori SIFT su una griglia regolare per ogni immagine, in seguito esegue

[3] `http://staff.science.uva.nl/~ksande/research/colordescriptors/readBinaryDescriptors.m`

[4] `http://www.cs.unc.edu/~lazebnik/research/SpatialPyramid.zip`

k-means per trovare il dizionario; al termine del processo a ciascun descrittore viene assegnata un'etichetta (*texton label*) corrispondente alla parola di dizionario più vicina. La piramide viene infine generata a partire da queste etichette. Ognuno degli step precedenti è stato assegnato ad una funzione specifica, in modo che ciascuna debba essere invocata in modo indipendente. Le funzioni sono:

- `GenerateSiftDescriptors`
- `CalculateDictionary`
- `BuildHistograms`
- `CompilePyramid`

Essendo le funzionalità di ricerca delle feature e costruzione del vocabolario già comprese nel codice che esegue il modello BoW, le uniche funzioni effettivamente modificate ed utilizzate per la costruzione della piramide spaziale sono `BuildHistograms` e `CompilePyramid`.

6.2.3 GIST

L'implementazione di GIST utilizzata è quella di Oliva e Torralba[5]. Il codice Matlab calcola il descrittore GIST per immagini quadrate: per questo ogni immagine caricata è stata ridimensionata al valore 256x256 sia per il dataset Caltech-101 che per MICC-Flickr-101. Il codice permette di impostare come parametri il numero di orientazioni per scala ed il numero di blocchi in cui dividere l'immagine.

6.2.4 pLSA-LDA

Per testare i due modelli a topic latenti introdotti nei precedenti capitoli si è fatto uso dell'implementazione realizzata nel 2006 da J.J. Verbeek[6]. L'autore fornisce alcuni script Matlab che consentono di determinare la distribuzione

[5] `http://people.csail.mit.edu/torralba/code/spatialenvelope/`
[6] `http://lear.inrialpes.fr/~verbeek/software.php`

dei topic per una serie di documenti dei quali siano note le occorrenze delle parole, fissati il numero di argomenti latenti desiderato ed alcuni parametri di configurazione. A seconda dei valori attribuiti ad alcuni di questi parametri, il codice eseguirà un pLSA o un LDA; essi consentono inoltre di stabilire se creare un modello ex-novo oppure usarne uno già calcolato, come si fa ad esempio durante la fase di test.
Nel dettaglio i parametri di configurazione sono i seguenti:

- `model`: se la variabile viene impostata, utilizza il modello fornito per il calcolo delle distribuzioni di topic;
- `learn_topics`: se 0, i topic non vengono aggiornati;
- `learn_alpha`: se 0, non vengono aggiornati i parametri delle Dirichlet dei topic;
- `learn_eta`: se 0, non vengon aggiornati i parametri delle Dirichlet delle parole;
- `var_beta`, `var_gamma`: se 0, si ottiene la versione di LDA *non-smooth*
- `var_gamma`: se 0 ed anche `var_beta`=0, si ottiene il modello pLSA;
- `alpha_sym`: se 1, la distribuzione a priori sui θ è simmetrica;
- `eta_sym`: se 1, la distribuzione a priori sui β è simmetrica.

I parametri binari, tutti tranne il modelo, vengono per default impostati a 1. La funzione `LDA_variational_inference`, dati i parametri in ingresso, il numero di topic e la distribuzione delle parole, fornisce in uscita il modello ottenuto e la distribuzione dei topic. Un esempio di parametri per la fase di addestramento è il seguente:

```
options.learn_topics=1;
options.learn_alpha=1;
options.learn_eta= 1;
options.var_beta=1;
options.var_gamma=1;
```

```
options.alpha_sym=0;
options.eta_sym=0;
```

In fase di test vengono aggiornati solo due parametri della variabile `options`:

```
options.model=model;
options.learn_topics=0;
```

con `model` il modello ottenuto al termine dell'apprendimento basato sulle immagini di train. Da notare che ponendo `learn_topics=0` non si modifica la distribuzione dei topic durante l'analisi degli elementi di test; lasciando invece sia `learn_alpha` che `learn_eta` ad 1 si consente di aggiornare gli iperparametri delle distribuzioni.
Per il corretto funzionamento degli script è necessario installare sul sistema le librerie `fastfit`[7] e `lightspeed`[8].

[7] `http://research.microsoft.com/en-us/um/people/minka/software/fastfit/`

[8] `http://research.microsoft.com/en-us/um/people/minka/software/lightspeed/`

Bibliografia

[1] Merriam-webster's collegiate dictionary. *10th ed. Springfield, Massachusetts, U.S.A*, 1994.

[2] L.von Ahn and L.Dabbish. Labeling images with a computer game. *CHI*, 2004.

[3] S.Narayanamurthy A.J.Smola. An architecture for parallel topic models. *PVLDB*, 2010.

[4] J.Sivic B.C.Russell, A.Efros. Using multiple segmentations to discover objects and their extent in image collections. *CVPR*, 2006.

[5] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[6] A. Bosch, A. Zissermann, and X. Munoz. Scene classification via plsa. *Proc. ECCV 2006*, 2006.

[7] A. Bosch, A. Zissermann, and X. Munoz. Scene classification using a hybrid generative/discriminative approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

[8] T. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng. Nus-wide: A real-world web image database from national university of singapore. July 8-10, 2009.

[9] C.Snoek, M.Worring, and A.Smeulders. Early versus late fusion in semantic video analysis. November 2005.

[10] E.Hörster, R.Lienhart, and M.Slaney. Image retrieval on large-scale image databases. *CIVR*, 2007.

[11] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. *CVPR*, 2005.

[12] Li Fei-Fei, R.Fergus, and P.Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *CVPR*, 2004.

[13] Li Fei-Fei, R.Fergus, and P.Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

[14] M. Girolami. Mercer kernel-based clustering in feature space. *IEEE Trans Neural Networks, 13(3),780 - 784*, 2002.

[15] K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features. *ICCV*, 2005.

[16] L.Ballan, M.Bertini, and A.Del Bimbo. Tag suggestion and localization in user-generated videos based on social knowledge. *WSM'10*, 2010.

[17] Li-Jia Li, R.Socher, and Li Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. *CVPR*, 2009.

[18] X. Li and C. Snoek. Visual categorization with negative examples for free. *ACM International Conference on Multimedia*, 2009.

[19] Xirong Li, C.G.M.Snoek, and M.Worring. Learning social tag relevance by neighbor voting. *IEEE Transactions on Multimedia*, 2009.

[20] Fei-Fei Li Liangliang Cao. Spatially coherent latent topic model for concurrent object segmentation and classification. *ICCV*, 2007.

[21] L.Kennedy, M.Slaney, and K.Weinberger. Reliable tags using image similarity. *WSMC*, 2009.

[22] D. G. Lowe. Object recognition from local scale-invariant features. *Proc. of the International Conference on Computer Vision*, September 1999.

[23] X. Ma, W. Eric, and L. Grimson. Learning coupled conditional random field for image decomposition with application on object categorization. *CVPR*, 2008.

[24] M.Andreetto, L.Zelnik-Manor, and P.Perona. Unsupervised learning of categorical segments in image collections. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2008.

[25] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 2005.

[26] M.J.Huiskes, B.Thomee, and M.S.Lew. New trends and ideas in visual concept detection: the mir flickr retrieval evaluation initiative. *MIR*, 2010.

[27] M.J.Huiskes and M.S.Lew. The mir flickr retrieval evaluation. *MIR*, 2008.

[28] N.Rasiwasia, J.Pereira, and E.Coviello. A new approach to cross-modal multimedia retrieval. *MM '10 Proceedings of the international conference on Multimedia*, 2010.

[29] O.Boiman, E.Shechtman, and M.Irani. In defense of nearest-neighbor based image classification. *CVPR*, 2008.

[30] A. Oliva and A. Torralba. Modeling the shape of a scene: a holistic representation of the spatial envelope. *IJCV 60*, 2004.

[31] A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. *Visual Perception, Progress in Brain Research*, 155, 2006.

[32] P.V.Gehler and S.Nowozin. On feature combination methods for multiclass object classification. *ICCV*, 2009.

[33] R.Lienhart, S.Romberg, and E.Hörster. Multilayer plsa for multimodal image retrieval. *CIVR*, 2009.

[34] F. Sebastiani. Machine learning in automated text categorization. *Journal of ACM Computing Surveys*, 2002.

[35] A. Setz and C. Snoek. Can social tagged images aid concept-based video search? *ICME'09 Proceedings of the 2009 IEEE international conference on Multimedia and Expo*, 2009.

[36] J. Sivic, B. Russel, A. Efros, A. Zissermann, and W.T. Freeman. Discovering objects and their locations in images. *ICCV*, 2005.

[37] S.Lazebnik, C.Schmid, and J.Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *CVPR*, 2006.

[38] T.Griffiths and M.Steyvers. Finding scientific topics. *PNAS April 6 and 2004 vol. 101 no. Suppl 1 5228-5235*, 2004.

[39] van de Sande, K. E. A., Gevers, T., Snoek, and C. G. M. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.

[40] J. van de Weijer, Th. Gevers, and A.D. Bagdanov. Boosting color saliency in image feature detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

[41] J.C. van Gemert, J.M.Geusebroek, C.J.Veenman, and A.W.M.Smeulders. Kernel codebooks for scene categorization. *ECCV*, 2008.

[42] J. Yang, Y. G. Jiang, A. G. Hauptmann, and C. W. Ngo. Evaluating bag-of-visual-words representations in scene classification. *Proc. of ACM International Workshop on Multimedia Information Retrieval (MIR)*, 2007.

[43] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao. Group-sensitive multiple kernel learning for object categorization. *IEEE International Conference on Computer Vision (ICCV)*, 2009.

[44] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. *Proc. of International Conference on Machine Learning (ICML)*, 1997.