

Kotlin Nutritional Facts Tables Detection

Visione Artificiale e Riconoscimento

Lisa Mazzini
Alberto Giunta

A.A. 2017/2018

Capitolo 1

Introduzione

1.1 Abstract

Con il problema dell'obesità che diventa sempre più attuale e importante, e con gli smartphone sempre più presenti all'interno della vita delle persone, è possibile oggigiorno considerare l'ipotesi di far passare anche l'educazione alimentare attraverso questi strumenti, al fine di sensibilizzare quante più persone possibile (specialmente i giovani) all'apprendimento di ciò che è bene e male per il nostro corpo e la nostra salute.

Questa *educazione* può passare ad esempio attraverso gli smartphone, che possono diventare quindi degli utili e pratici strumenti per capire cosa stiamo comprando o mangiando. Ad esempio puntando la fotocamera del device verso la confezione dell'alimento, questo può comunicarci in termini più esplicativi cosa significano per la nostra dieta i valori osservati.

Quello che presenteremo qui di seguito è quindi un algoritmo per la localizzazione di coppie "Proprietà Nutrizionale" - "Valore Nutrizionale" all'interno di Tabelle Nutrizionali, prese a partire da immagini scattate con smartphone off-the-shelf.

L'algoritmo ha l'obiettivo di effettuare il miglior pre-processing possibile per ottenere un'immagine pulita, non distorta, e dimensioni ottimali per poter effettuare la scansione OCR ed ottenere i valori della tabella analizzata.

Il pre-processing prevede inoltre l'utilizzo di filtri, di binarizzazione e della trasformata di Hough per individuare linee all'interno dell'immagine che potrebbero rappresentare la tabella, e per raddrizzare quanto più possibile la foto, al fine di facilitare il lavoro all'algoritmo di Optical Character Recognition.

1.2 Descrizione del problema

Come descritto nell'abstract, con questo progetto si è cercato un modo di facilitare il percorso verso la creazione di applicativi in grado di educare ed assistere le persone nella consultazione delle Tabelle Nutrizionali presenti sulle confezioni dei prodotti venduti nei supermercati.

La necessità di applicativi come questi potrebbe ad esempio nascere da diverse motivazioni, ad esempio la scarsa conoscenza del significato e dei reali riscontri sulla propria salute di ciò che è scritto all'interno delle Tabelle Nutrizionali, oppure il fatto che i diversi packaging (colori, font particolari) possono risultare di difficile consultazione da parte di persone anziane.

Idealmente questi sistemi andrebbero implementati in forma di applicazioni da installare sul proprio smartphone. Ciò che è stato prodotto per questo progetto è Desktop based, ma è completamente compatibile con i sistemi Android. Lo sforzo necessario per il porting su questo sistema operativo si limiterebbe all'interfacciamento del sistema attuale con le API di sistema più specifiche, ad esempio per quanto riguarda la gestione della fotocamera o dei processi.

Per la risoluzione di questo problema sono stati imposti innanzitutto i vincoli del dominio applicativo di cui ci si occupa, tra cui:

- Sono stati considerati solo prodotti in italiano con Tabelle Nutrizionali scritte solo o anche in Italiano.
- Sono state considerate solo le Tabelle Nutrizionali più comuni, ovvero quelle in forma tabellare verticale e con due sole colonne, con i nomi delle proprietà (grassi, proteine, carboidrati, ecc) sulla sinistra, e i loro rispettivi valori sulla destra. Non sono state considerate quindi tabelle scritte "in prosa" o su più colonne, o in orizzontale.
- Si è data maggiore priorità al permettere alla libreria di OCR di poter riconoscere quanto più testo possibile, piuttosto che alla correttezza del testo riconosciuto. Questo perchè virtualmente in tal modo è stato possibile astrarre il funzionamento dell'algoritmo dalla specifica libreria, che un giorno potrà essere sostituita con un algoritmo più potente e robusto in termini di variabilità di font e caratteri.
- Le Tabelle Nutrizionali per prodotti venduti in Italia devono essere omologate a una struttura pressochè predefinita, ovvero devono prevedere forzatamente le voci per: "energia", "grassi", "acidi grassi saturi", "fibre", "proteine", "sale".
- Si presuppone che l'immagine fornita all'algoritmo non sia esageratamente ruotata rispetto all'asse verticale, sono ammesse però rotazioni di circa 10.

1.3 Obiettivo del progetto

Come già brevemente descritto sopra, l'obiettivo finale di questo progetto è quello di produrre un algoritmo in grado di individuare la tabella nutrizionale all'interno di una immagine, predisponendola in maniera ottimale per far sì che l'engine di OCR fosse in grado di individuare le proprietà e i valori ad esse correlate, per poi farne il pairing.

Si è deciso quindi di non considerare come cattivi risultati valori mal letti dall'OCR. Piuttosto ci si è concentrati sulla corretta localizzazione delle informazioni di rilievo, in modo tale da facilitare sviluppi futuri con tecnologie più potenti, che facilmente possono essere sostituite a quelle usate attualmente.

1.4 Tecnologie

L'implementazione dell'algoritmo è JVM based, scritto in Kotlin, un nuovo linguaggio misto funzionale sviluppato da Jetbrains e recentemente direttamente supportato anche da Google.

Per quanto riguarda gli algoritmi di Computer Vision è stata utilizzata JavaCV[1], un porting di OpenCV per JVM.

Relativamente invece all'engine che si doveva occupare di OCR la scelta è ricaduta su Tess4j[2], che è la rappresentazione per JVM di Tesseract, una libreria sviluppata e mantenuta da Google e probabilmente la migliore nel mondo Open Source.

Tutte le fotografie utilizzate per la validazione dell'algoritmo e per i test sono state scattate su smartphone, tra cui:

- LG Nexus 5X - 12.3 Mpx
- Samsung S5 Neo - 16 Mpx

Capitolo 2

Studi correlati

In letteratura sono presenti diversi studi riguardanti l'individuazione e la lettura di tabelle all'interno di immagini, siano queste scansioni di documenti o fotografie di prodotti alimentari. In particolare sono stati individuati due studi affini all'obiettivo del progetto e verranno in seguito riassunti brevemente.

On Mobile Detection and Localization of Skewed Nutrition Facts Tables - Christopher Blay, 2013[3]

L'obiettivo dello studio è stabilire se l'immagine che si sta analizzando contiene o meno una tabella nutrizionale, anche se ruotata fino a 30 gradi in qualsivoglia direzione, e in caso affermativo individuare la sua posizione. La precondizione necessaria per il corretto funzionamento dell'algoritmo è che l'unico elemento a fuoco all'interno dell'immagine sia la tabella nutrizionale.

Il primo passo della tecnica proposta è correggere un'eventuale rotazione della tabella presente all'interno della foto: l'immagine viene convertita in greyscale, ne vengono individuati gli edges tramite l'algoritmo Canny Edge Detection e su questi viene applicata la trasformata di Hough. Questa individua le principali rette presenti all'interno dell'immagine e calcola l'angolo medio di deviazione rispetto agli assi principali, il quale verrà poi utilizzato per applicare la rotazione all'immagine.

Una volta ottenuta l'immagine correttamente allineata, utilizza il metodo Dilate/Erode Corner Detection per individuare i corner presenti al suo interno. Poichè si assume che l'unico testo a fuoco, quindi ben definito, sia quello della tabella nutrizionale, l'area dell'immagine con una maggiore presenza di corner sarà quella con la maggior quantità di testo, ovvero quella contenente la tabella. Tutto ciò che ha una densità minore di corner viene considerato come sfondo.

Per delimitare i bordi esatti della tabella vengono sommati i pixel individuati come corner per righe e per colonne. L'intervallo di righe e colonne con la somma superiore a una certa soglia stabilisce individua i margini esterni della tabella nutrizionale. Per ulteriore conferma, l'autore replica il preprocessing necessario per la trasformata di Hough e cerca la linea individuata da essa che si avvicina maggiormente ai margini individuati in precedenza.

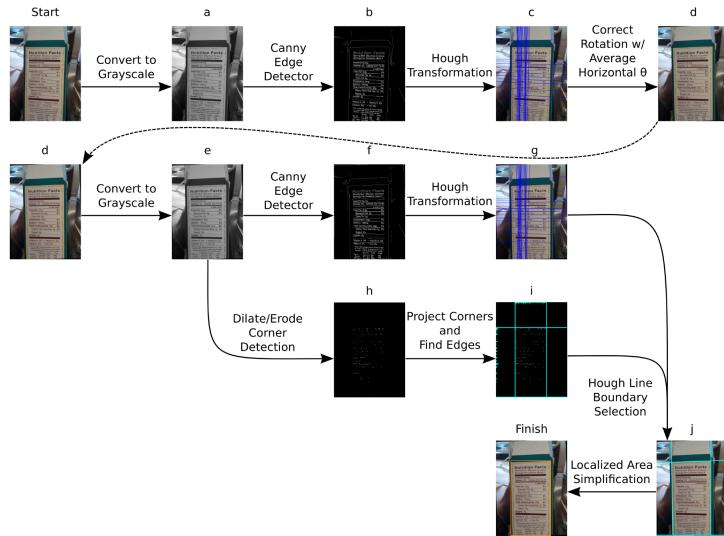


Figura 2.1: Schema riassuntivo dei passi dell'algoritmo proposto in [3]

L'algoritmo presenta alcuni limiti in quanto è efficace solo su immagini contenenti esclusivamente la tabella e non testo estraneo, tabella che inoltre deve essere in forma rettangolare e priva di deformazioni. I risultati ottenuti sono riportati in figura.

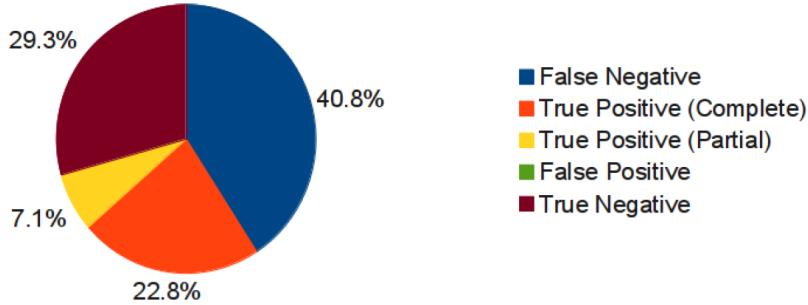


Figura 2.2: Risultati della localizzazione

Precision	Total Recall	Complete Recall	Partial Recall	Specificity	Accuracy
0.7632	0.4222	0.3580	0.1475	1.0	0.5916

Figura 2.3: Valori di performance della localizzazione

Recognition of Nutrition Facts Labels from Mobile Images - Olivia Grubert, Linyi Gao, 2014[4]

A differenza del paper precedentemente discusso, l'obiettivo di questo studio non si limita a individuare la tabella nutrizionale ma mira anche alla comprensione dei valori riportati al suo interno. La precondizione necessaria al corretto funzionamento dell'algoritmo è che ciascuna riga della tabella sia delimitata da linee rette.

La prima fase dell'algoritmo prevede il preprocessing dell'immagine con l'obiettivo di renderla più chiara possibile una volta binarizzata, quindi si applicano una serie di filtri di smoothing e di equalizzazione dell'istogramma in modo da eliminare l'eventuale glare all'interno dell'immagine. L'immagine viene quindi binarizzata e, poiché l'obiettivo è quello di individuare le linee presenti all'interno della tabella, vengono rimosse tutte le aree di dimensione ridotta rilevate.



Figura 2.4: Preprocessing dell'immagine

Vengono poi clusterizzate quelle con orientazione simile in modo da non prendere in considerazione le linee non appartenenti alla tabella. Il passaggio successivo è un eventuale rotazione dell'immagine se non perfettamente allineata tramite trasformata di Hough.

Lo step successivo è quello di ritagliare le singole righe della tabella, contenenti la proprietà con il rispettivo valore. Viene fatto ciò in quanto l'algoritmo di OCR è risultato più efficace se operante sulle singole linee piuttosto che sull'intera tabella come mostrato in figura 2.5.

Per effettuare l'OCR delle immagini è stato utilizzato Tesseract[5]. L'output restituito viene poi ulteriormente processato confrontandolo con un dizionario di parole note, calcolando la distanza di Levenshtein. In tal modo vengono corretti gli eventuali errori commessi da Tesseract e aumentare di gran lunga la correttezza dell'algoritmo, abbassando la percentuale di errore dal 53,3% al 36,2% su un dataset di 97 immagini.

% Daily Value*		
Total Fat 1g*	2%	2%
Saturated Fat 0g	0%	0%
Trans Fat 0g		
Polynsaturated Fat 0g		
Monounsaturated Fat 0g		
Cholesterol 0mg	0%	0%
Sodium 210mg	9%	12%
Potassium 90mg	3%	8%
Total Carbohydrate 44g	15%	17%
Dietary Fiber 3g	12%	12%
Sugars 14g		
Protein 4g		

Total Fat 1g*	2%	2%
Saturated Fat 0g	0%	0%

Figura 2.5: Output di Tesseract relativo all'intera tabella (in alto) e relativo alle singole righe (in basso)

Altri studi

Sono stati poi analizzati altri studi che però si allontanano in parte dall'obiettivo del progetto in esame. Diversi paper ad esempio riportano algoritmi finalizzati alla detection di tabelle all'interno di un documento, spesso scannerizzato.

Uno di questi, *Detection of Table Structure and Content Extraction From Scanned Documents*[6] procede applicando degli operatori morfologici per rilevare all'interno della pagina le linee della tabella; grazie alle linee verticali e orizzontali individuate, è possibile trovare le coordinate delle singole celle e quindi estrarre il contenuto. Tuttavia il contenuto viene estratto sempre sottoforma di immagine, senza applicare alcun algoritmo di OCR.

Un algoritmo simile è descritto all'interno di *Table Frame Line Detection in Low Quality Document Images Based on Hough Transform*[7] dove, dopo aver applicato una correzione in caso di rotazione della tabella, si ricercano le linee che delimitano la tabella tramite un algoritmo di Run Length Smoothing (RLSA) e ne studiano in particolare anche le diverse tipologie di intersezioni, sfruttando la trasformata di Hough. La stessa trasformata, in unione con l'algoritmo di Harris per la corner detection, è utilizzata anche in *A Multi Clue Heuristic Based Algorithm For Table Detection*[8]; i risultati dei due algoritmi vengono poi matchati tra loro tramite un nearest neighbor framework.

Tuttavia gli algoritmi sopra proposti si sono rivelati troppo limitati, in quanto si presuppone sempre che la tabella sia sempre delimitata da linee nette sia orizzontali che verticali, condizione non sempre vera nel dominio applicativo del progetto proposto.

Verranno dunque sfruttati maggiormente i primi due paper sopra riportati.

Capitolo 3

Tecnica proposta

3.1 Image preprocessing

3.1.1 Preprocessing per la rotazione

La prima fase del preprocessing prevede un'analisi dell'immagine di input con l'obiettivo di ruotarla per allineare il testo e al tabella con l'asse X.

In questa sezione verranno descritti i passi che vengono applicati ognqualvolta si decide di applicare la trasformata di Hough sull'immagine data.

1. **Resize:** si applica il resize dell'immagine con un fattore di 0.5X. Questo viene fatto poichè è emerso che la trasformata di Hough risulta più accurata se effettuata su immagini di dimensioni ridotte.
2. **Grayscale:** è necessario per applicare una riduzione di informazione sull'immagine, in vista dell'applicazione di algoritmi che potrebbero essere tratti in inganno dalla presenza di colore.
3. **Contrast:** dal momento che si è notato che la stragrande maggioranza di Tabelle Nutrizionali sono scritte in inchiostro scuro, è utile applicare un leggero livello di contrasto che permette di aumentare la differenza di intensità tra testo (e linee della tabella) e lo sfondo (spesso più chiaro), affinché sia possibile in seguito ottenere una binarizzazione più accurata.
4. **Sobel Filter:** necessario per evidenziare tutti i bordi che risultano interessanti nell'applicazione della trasformata di Hough.
5. **Binarizzazione Otsu:** è l'ultimo step prima di applicare Hough. Applicando la binarizzazione si fa in modo da creare un'immagine con l'informazione necessaria alla trasformata di Hough al fine di trovare righe e linee all'interno dell'immagine.
6. **Trasformata di Hough:** si utilizza la trasformata di Hough per il modello retta per individuare le linee all'interno dell'immagine, con una accuratezza tale da poterle poi usare per raddrizzare l'immagine stessa rispetto agli assi X e Y. Per ottenere un buon numero di rette su cui basare i successivi ragionamenti, si applica ricorsivamente la trasformata di Hough partendo da un numero minimo di voti per ogni retta pari a 1000, e riducendolo poi di uno step predefinito a ogni esecuzione. L'iterazione si interrompe quando viene trovato un numero sufficiente di rette sia verticali che orizzontali.

La necessità di fare ciò si è presentata poiché non è stato possibile stabilire a priori un numero minimo di voti che fosse adatto all'individuazione di un sufficiente numero di rette in ogni immagine di input. Con un numero fisso di voti minimi infatti per alcune immagini la trasformata avrebbe trovato un numero eccessivo di rette, che avrebbero poi "sporcato" l'angolo medio di inclinazione. Una volta ottenuto un numero sufficiente di linee queste vengono poi ulteriormente filtrate, al fine di tenere solo quelle la cui inclinazione ricade all'interno di un range prestabilito (circa 10 gradi) rispetto al proprio asse (orizzontale o verticale).

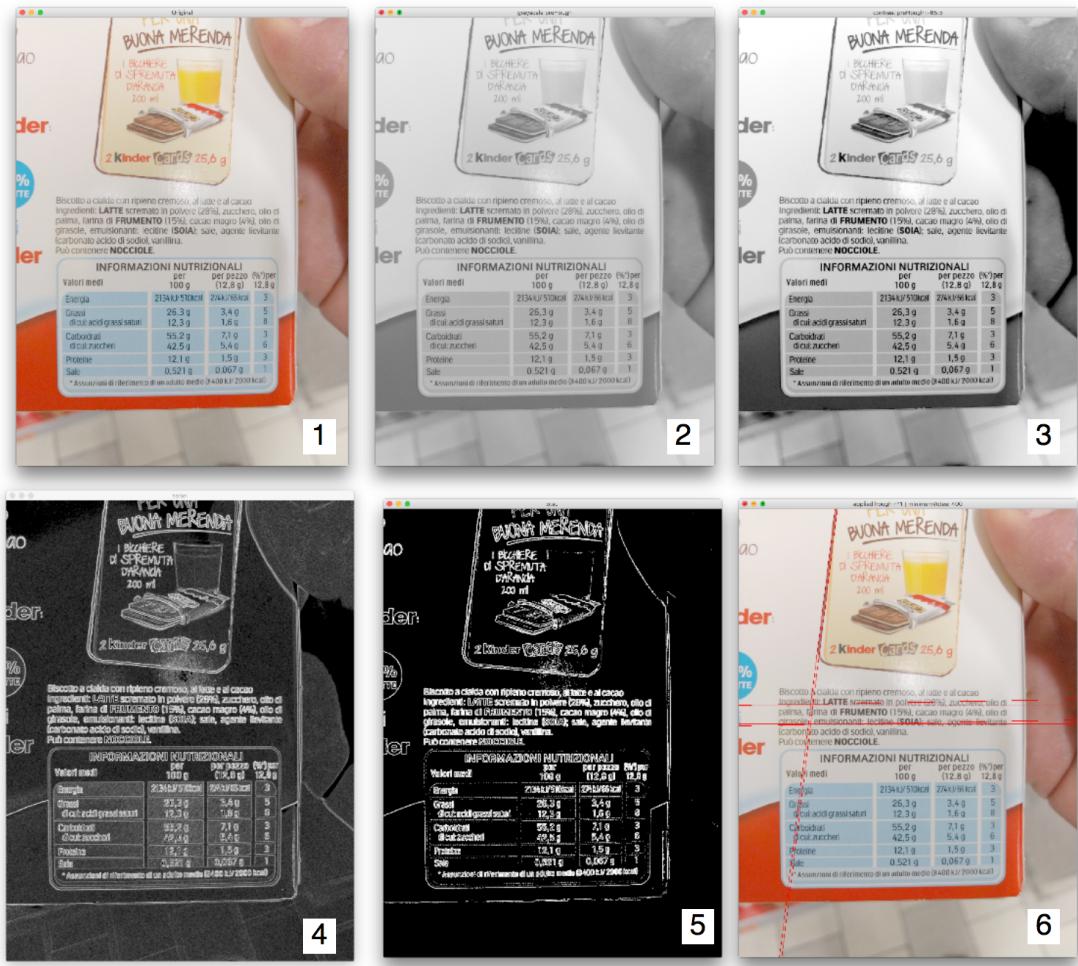


Figura 3.1: Fasi del preprocessing finalizzato alla rotazione

3.1.2 Rotazione

Grazie ai passi elencati nella sezione 3.1.1 è ora possibile calcolare il Θ , ovvero l'angolo di rotazione dell'immagine affinchè la tabella e le scritte siano allineate rispetto all'asse X.

Questo parametro viene calcolato come una media dei Θ delle linee orizzontali e verticali relativi a rispettivi assi.

La rotazione ovviamente si applica all'immagine originale, e non più a quella usata per il preprocessing per Hough.

L'immagine originale ruotata verrà quindi inoltrata alla prossima fase di preprocessing, il cui obiettivo è quello di rimuovere tutto il background a sinistra e sopra alla tabella. Si esegue infatti l'algoritmo di OCR per individuare alcune parole chiave che delimitano i margini superiore e a destra della tabella stessa.

3.1.3 Preprocessing per OCR

Per ottimizzare l'immagine originale al fine di eseguire l'OCR vengono applicati una serie di passi:

1. **Riduzione colore:** vengono ridotti il numero di colori codificati all'interno dell'immagine, per ridurre la quantità di informazioni non rilevanti ai fini dell'OCR.
2. **Grayscale:** come per la riduzione colore, l'obiettivo è quello di perdere quanta più informazione inutile possibile.
3. **Contrasto:** si vogliono rendere più evidenti il testo e le linee rispetto allo sfondo.
4. **Binarizzazione Global Threshold:** viene applicata in questo caso una binarizzazione con soglia globale, determinata dopo una serie di test empirici. Qui non è stato utilizzato Otsu per diversi motivi: il primo è il fatto che la soglia dinamica di Otsu veniva molto influenzata da eventuali zone molto scure all'interno dell'immagine (ad esempio nei margini neri introdotti dalla rotazione dell'immagine) e causavano una binarizzazione non corretta nelle zone di testo che non ne permetteva la lettura. Inoltre si è notato che la soglia trovata da Otsu non era ottimale per le immagini con testo su sfondo colorato, ma era efficace solo su immagini testo nero su sfondo bianco. Tramite la soglia globale quindi è stato possibile trovare un valore che potesse funzionare bene su tutte le immagini, considerando che l'unica zona dell'immagine di cui era necessario preservare l'informazione è quella della Tabella Nutrizionale.

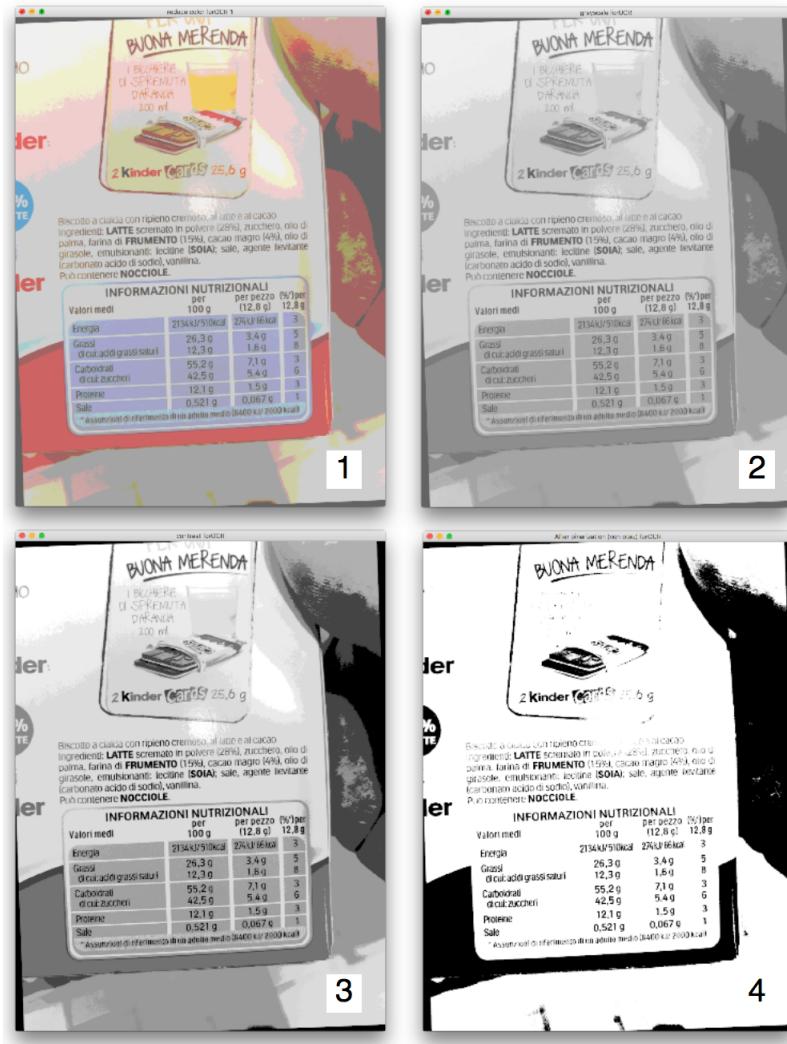


Figura 3.2: Fasi del preprocessing finalizzato al primo OCR

3.1.4 OCR

Ottenuta l'immagine ruotata e binarizzata si può eseguire l'OCR dell'immagine a dimensione originale. L'obiettivo di questo passo è trovare dei keypoint rappresentati da specifiche parole presenti all'interno della tabella, che permetterebbero di cropare l'immagine sul lato superiore e sul lato sinistro. In pratica quello che è stato fatto è:

- Sono stati creati due dizionari (come si può vedere alla sezione 6.1), uno per le parole utilizzabili per il crop sul lato superiore (ad esempio "nutrizionale", che è sempre presente all'interno del titolo della Tabella), e uno per il crop sul lato sinistro (contenente parole come "carboidrati" o "proteine" che sono parole sempre contenute nelle Tabelle e sono sempre posizionate a sinistra).
- Si calcola la distanza di Levenshtein tra ogni parola che viene trovata dall'OCR e le parole contenute in ciascun dizionario.
- Si salvano solo le parole con distanza minore da quelle dei rispettivi dizionari, e si fa un crop "cauto", ovvero, per non perdere porzioni di tabella, tra le parole salvate si prendono solo quelle più in alto, o quelle più a sinistra di tutte.

3.1.5 Crop

Il crop dell'immagine è un passo estremamente necessario ai fini dell'OCR finale in cui verranno trovate le proprietà nutritive ed i loro valori. Il crop dell'immagine permette infatti di ridurre notevolmente la quantità di informazioni superflue (tutto ciò che è a sinistra o sopra la tabella nutrizionale scompare), e si cerca di isolare quanto più possibile la tabella.

La procedura per isolare la tabella è parziale, infatti ciò che è collocato alla sua destra o sotto di essa non viene tagliato. Questo perché è complesso determinare l'angolo in basso a destra della tabella, specialmente in tabelle non delineate da righe, in quanto non si può stabilire a priori la larghezza o l'altezza della tabella, nè quale sarà l'ultima proprietà contenuta al suo interno.

L'immagine croppata sarà quella che verrà poi utilizzata per i prossimi passi di preprocessing che verranno ora elencati.

3.1.6 Preprocessing per Localizzazione e OCR

I prossimi passi di preprocessing verranno applicati sull'immagine originale ma tagliata e permetteranno di avere un'immagine predisposta e adatta all'individuazione delle proprietà nutritionali e dei rispettivi valori.

I passi che vengono eseguiti sono quindi:

1. **Preprocessing per Rotazione:** vengono applicati nuovamente i passi descritti nella sezione 3.1.1
2. **Rotazione:** vengono applicati nuovamente i passi descritti nella sezione 3.1.2. Grazie al parziale isolamento della tabella, ci saranno meno informazioni che possono influenzare negativamente la trasformata di Hough, e quindi la rotazione sarà ancora più accurata rispetto all'iterazione precedente.
3. **Preprocessing per OCR:** vengono applicati nuovamente i passi descritti nella sezione 3.1.3



Figura 3.3: La prima immagine mostra il risultato dei passi al punto 1., la seconda mostra la correzione della rotazione applicata come descritto al punto 2., la terza immagine è quella ottenuta dopo i passi al punto 3., e verrà utilizzata per effettuare l'OCR per estrarre proprietà e valori annessi.

3.2 Localizzazione e OCR

3.2.1 Individuazione delle proprietà nutrizionali

L’immagine preprocessata e ritagliata viene poi passata a Tesseract per applicare l’OCR, restituendo una lista di Word, oggetti contenenti il testo individuato, la bounding box che lo racchiude e la confidenza con cui il testo è stato letto. È stato poi creato un dizionario di parole tipicamente presenti all’interno di una tabella nutrizionale per individuare le proprietà nutrizionali, come mostrato in tabella in 6.1.

Per ogni parola individuata da Tesseract viene calcolata la distanza di Levenshtein con ogni parola del dizionario. Vengono poi filtrate solo le parole che ottengono una distanza inferiore a una certa soglia, in modo da eliminare tutte le parole estranee al dominio della tabella. Poichè è possibile che più parole abbiano matchato con una sola di quelle del dizionario, per evitare di avere ripetizioni, viene mantenuta solo la parola che ha ottenuto distanza minore.

Alcune proprietà sono composte da più parole (es. ”acidi grassi saturi”), quindi è stato necessario un lavoro di merging tra le parole distinte individuate ma la cui bounding box fosse circa sulla stessa coordinata y (ovvero allineate sulla stessa riga). Durante il merging, viene ricostruita la bounding box tenendo conto delle dimensioni congiunte delle due o più bounding box di partenza.

Infine, per un’ulteriore conferma di aver individuato solo le proprietà nutrizionali presenti all’interno della tabella, viene calcolata la coordinata X media delle parole individuate con maggiore certezza, e si filtrano soltanto le parole allineate sulla stessa X (con un margine di errore), come da layout generale della tabella.

3.2.2 Individuazione dei valori numerici

Per rilevare i valori numerici relative alle proprietà non si può per ovvie ragioni fare riferimento a un dizionario, per cui l'algoritmo procede solo basandosi sulla loro posizione.

Come prima fase di filtering, dalle parole individuate viene rimosso tutto ciò che non contiene almeno un numero. In seguito vengono eliminate le parole che si trovano a sinistra della proprietà nutrizionale più a destra.

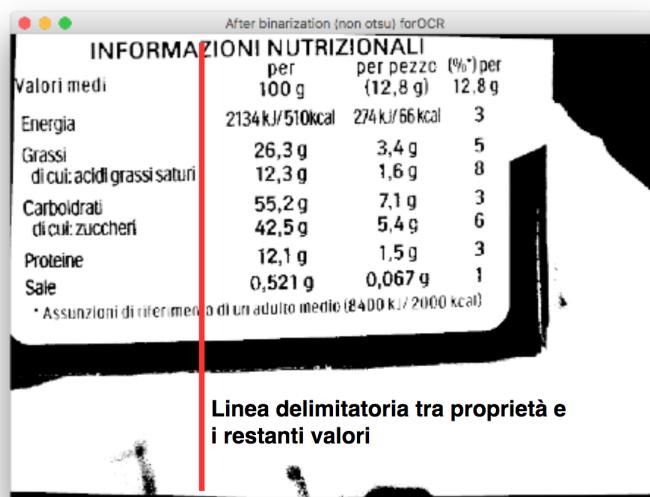


Figura 3.4: Esempio di filtraggio dei valori

È emerso durante l'esecuzione dell'algoritmo di OCR che molto frequentemente Tesseract interpreta erroneamente le *g* di grammi come numeri, spesso 9 o 8, su singole parole. Per cui si è deciso di controllare per ogni valore individuato se la parola successiva, nel caso fosse un 9 o un 8, fosse sulla stessa linea (quindi con una coordinata *y* simile). In caso affermativo, si procede eliminando la parola composta solo da 9 o 8, e di aggiungere la *g* di grammi al valore preso in considerazione.

Al termine di questa fase si hanno quindi tutti i termini contenenti almeno un numero, a destra delle proprietà ed eventualmente con l'unità di misura dei grammi, se non già correttamente individuata.

3.2.3 Merging tra proprietà e valori nutrizionali

Il passo successivo prevede un analisi di ogni proprietà con ogni elemento che ha superato la fase precedente di filtering.

Per ogni proprietà si confrontano tutti gli elementi e vengono estratti quelli allineati sulla stessa Y della proprietà stessa; di questi, verrà conservato solo quello con la X minore, quindi più vicino alla proprietà in esame.

In questo modo viene creata un'associazione tra proprietà e valore solo nel caso in cui risponda alle caratteristiche di posizione sopra descritte. Se nessun valore si allinea alla proprietà, questa rimarrà senza valore associato. Per mostrare a video i risultati ottenuti, per ogni proprietà e valore individuati viene disegnata la relativa bounding box sull'immagine originale.

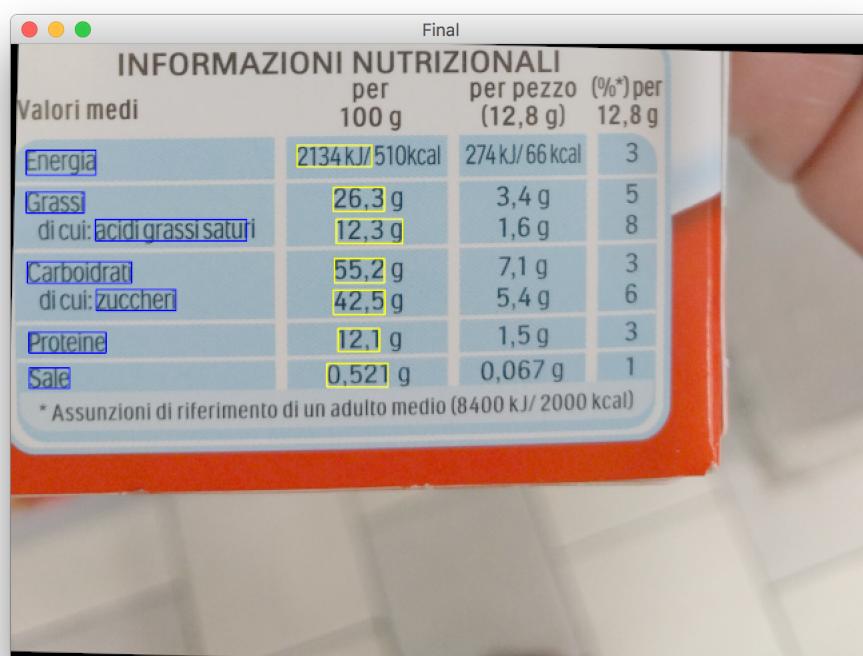


Figura 3.5: Risultato finale del merging tra proprietà (in blu) e valori(in giallo)

Capitolo 4

Risultati

4.1 Set utilizzati

I set utilizzati contengono immagini raccolte appositamente per questo progetto. È stata fatta questa scelta poiché per il corretto funzionamento dell'algoritmo erano necessarie tabelle nutrizionali in italiano strutturate in uno specifico modo; è stato dunque più efficace formare il set di immagini manualmente.
I set contengono in tutto N immagini.

4.1.1 Validation Set

Il validation set utilizzato è composto da 42 immagini di tabelle nutrizionali di diverso tipo. I risultati ottenuti in seguito al tuning dei parametri¹ dell'algoritmo sono riportati nella tabella sottostante.

Set	Corretti(completi)	Corretti(parziali)	Errati
Training	63,1 %	11,0%	25,9%

4.1.2 Test Set

Il test set è formato da 25 immagini di tabelle nutrizionali di diverso tipo. Con i parametri fissati in fase di tuning i risultati ottenuti sono riportati nella tabella sottostante.

Set	Corretti(completi)	Corretti(parziali)	Errati
Test	60,4%	14,7%	24,9%

¹I parametri utilizzati sono riportati nell'Appendice 6.2

4.2 Commenti sui risultati ottenuti

È stato notato che l'algoritmo riesce a ottenere risultati tanto migliori quanto più nitido è il testo rispetto allo sfondo. Ci sono stati infatti dei casi di tabelle con uno sfondo a righe a colori alternati che rendono difficile se non impossibile una corretta binarizzazione dell'immagine.

Energia	316 kJ - 75 kcal	Energia	316 kJ - 75 kcal
Grassi	2,0 g	Grassi	2,0 g
di cui acidi grassi saturi	0,1 g	di cui acidi grassi saturi	0,1 g
Carboiodrati	8,4 g	Carboiodrati	8,4 g
di cui zuccheri	1,8 g	di cui zuccheri	1,8 g
Fibre	4,7 g	Fibre	4,7 g
Proteine	3,6 g	Proteine	3,6 g
Sale	0,12 g	Sale	0,12 g

Figura 4.1: Esempio di tabella a sfondo con righe di colore alternato

Altri casi di problematiche che sono stati riscontrati ricadono sempre nelle immagini che usano font o tecniche di stampa particolari, che lasciano del rumore sullo sfondo che intralcia l'OCR.

DICHIAZIONE NUTRIZIONALE per 100 ml NÄHRWERTDEKLARATION je 100 ml	
Energia - Energie	204 kJ/48 kcal
Grassi - Fett	1,6 g
di cui acidi grassi saturi - davon gesättigte Fettsäuren	1,2 g
Carboiodrati - Kohlenhydrate	4,9 g
di cui zuccheri - davon Zucker	4,8 g
Proteine - Eiweiß	3,6 g
Sale - Salz	0,10 g
Calcio - Kalzium	120 mg = 15%**
Lattosio - Laktose < 0,01 g	
**15% NRV = Valori nutritivi di riferimento - Nährstoffbezugswerte	

DICHIAZIONE NUTRIZIONALE per 100 ml NÄHRWERTDEKLARATION je 100 ml	
Energia - Energie	204 kJ/48 kcal
Grassi - Fett	1,6 g
di cui acidi grassi saturi - davon gesättigte Fettsäuren	1,2 g
Carboiodrati - Kohlenhydrate	4,9 g
di cui zuccheri - davon Zucker	4,8 g
Proteine - Eiweiß	3,6 g
Sale - Salz	0,10 g
Calcio - Kalzium	120 mg = 15%**
Lattosio - Laktose < 0,01 g	
**15% NRV = Valori nutritivi di riferimento - Nährstoffbezugswerte	

Figura 4.2: Esempio di immagine con stampa a grana particolarmente grossa

INFORMAZIONI NUTRIZIONALI Valori medi per 100 g di prodotto		
ENERGIA	kj (kcal)	2148 (514)
GRASSI VEGETALI	g	26
di cui ACIDI GRASSI SATURI	g	2,5
di cui ACIDI GRASSI MONOINSATURI	g	22
di cui ACIDI GRASSI POLINSATURI	g	1,5
CARBOIDRATI	g	57
di cui ZUCCHERI	g	45
FIBRE ALIMENTARI	g	6
PROTEINE	g	10
SALE	g	0,001

Figura 4.3: Esempio di immagine con font particolare

Per quanto riguarda l'OCR in particolare, si sono riscontrati dei problemi nel riconoscimento dei valori nutrizionali (quelli a destra delle proprietà). Frequentemente infatti l'algoritmo ha erroneamente interpretato le unità di misura in grammi come dei "9" o "8". Inoltre, le virgolette spesso non vengono individuate.

Si suppone che questo genere di problemi sia risolvibile utilizzando un engine di OCR più potente, o meglio addestrato, in quanto il preprocessing delle immagini risulta essere il più delle volte corretto o quantomeno accettabile.

ALORI NUTRI- ONALI MEDI DI BRODO	Per 100 ml	Per porzione**	%* per porzione**	ALORI NUTRI- ONALI MEDI DI BRODO	Per 100 ml	Per porzione**	%* per porzione**
allore energetico	20 kJ 4 kcal	50 kJ 10 kcal	<1%	allore energetico	20 kJ 4 kcal	50 kJ 10 kcal	<1%
Grassi	0 g	0 g	<1%	Grassi	0 g	0 g	<1%
di cui saturi	0 g	0 g	<1%	di cui saturi	0 g	0 g	<1%
Carboidrati	0,7 g	2,0 g	<1%	Carboidrati	0,7 g	2,0 g	<1%
di cui zuccheri	0,2 g	0,5 g	<1%	di cui zuccheri	0,2 g	0,5 g	<1%
Proteine	0,4 g	1,0 g	2%	Proteine	0,4 g	1,0 g	2%
Sale	1,2 g	3,0 g	50%	Sale	1,2 g	3,0 g	50%

energetico	2010
grassi	0
saturi	0
carboidrati	079
proteine	0,49
sale	1,29

Figura 4.4: Esempio di immagine dove vengono confusi le "g" dei grammi con dei "9", e dove le virgolette non vengono talvolta lette

Capitolo 5

Conclusioni

5.1 Problemi riscontrati

Il problema principale è stato probabilmente quello della grave mancanza di documentazione all'interno delle librerie che sono state utilizzate. Entrambe (JavaCV e Tess4j) sono null'altro che dei porting su JVM di librerie nate per essere utilizzate in C++.

Un altro dei problemi riscontrati, oltre alla mancanza di documentazione, è che il porting non è sicuramente stato fatto nella maniera più idiomatica possibile, costringendo i programmatori Java ad usare API talvolta contro-intuitive poiché derivanti dall'implementazione C++ sottostante.

Per quanto riguarda l'algoritmo più nello specifico, nell'ambito di Computer Vision all'interno del pre-processing sono stati riscontrati alcuni problemi con immagini:

- non perfettamente a fuoco
- con testo scuro su sfondi scuri o viceversa con testo chiaro su sfondi chiari
- con packaging proni a deformazioni, come pacchi di farina, sacchetti ecc.

L'engine di OCR Tesseract ha inoltre delle difficoltà quando i font si allontanano da quelli più comuni e regolari. Dove i font e i design diventano infatti creativi o particolarmente unici, c'è una grossa perdita di informazione causata dalla rarità di questi ultimi, e dal loro essere facilmente fraintesi.

5.2 Sviluppi futuri

Come già anticipato, l'implementazione è stata pensata per astrarre quanto più possibile dalle tecnologie e dalle librerie utilizzate, in maniera tale da risultare facilmente intercambiabili in futuro.

Un buon esempio potrebbe essere quello di utilizzare le Cloud Vision API di Google o l'engine Abbyy al posto di Tesseract per quanto riguarda l'OCR.

Un altro obiettivo è sicuramente quello di effettuare il porting dell'algoritmo su sistemi mobile (Android e iOS), dove per Android lo sforzo dovrebbe risultare minimo rispetto ad iOS, essendo l'implementazione già JVM based, con librerie egualmente disponibili sia per l'ambiente desktop che per quello mobile.

Bibliografia

- [1] *JavaCV, Java interface to OpenCV, FFmpeg, and more*
<https://github.com/bytedeco/javacv>
- [2] *Tess4j, A Java JNA wrapper for Tesseract OCR API,*
<http://tess4j.sourceforge.net/>
- [3] Christopher Blay, *On Mobile Detection and Localization of Skewed Nutrition Facts Tables*, Utah State University, 2013.
- [4] O. Grubert, L. Gao *Recognition of Nutrition Facts Labels from Mobile Images* , Stanford University, 2014
- [5] *Tesseract Optical Character Recognition engine*,
<https://opensource.google.com/projects/tesseract>
- [6] S.Deivalakshmi, K.Chaitanya and P.Palanisamy *Detection of Table Structure and Content Extraction From Scanned Documents* , International Conference on Communication and Signal Processing, 2014
- [7] Y. Tian, C. Gao, X. Huang *Table Frame Line Detection in Low Quality Document Images Based on Hough Transform*, 2nd International Conference on Systems and Informatics 2014
- [8] R Naganjaneyulu, N Veerendra Sathwik, A.V. Narasimhadhan *A Multi Clue Heuristic Based Algorithm For Table Detection*, IEEE Region 10 Conference, 2016

Capitolo 6

Appendice

6.1 Dizionari

Dizionario	Parole contenute
Properties	”energia”, ”energetico”, ”grassi”, ”acidi”, ”saturi”, ”insaturi”, ”monoinsaturi”, ”polinsaturi”, ”carboidrati”, ”zuccheri”, ”proteine”, ”fibre”, ”sale”, ”fibra”
X	”informazioni”, ”tabella”, ”dichiarazione”, ”nutrizionale”, ”nutrizionali”
Y	”energia”, ”grassi”, ”carboidrati”, ”proteine”, ”sale”

6.2 Parametri

6.2.1 Contrasto

Parametro	Valore
Desired depth of the output matrix	rtype = -1 (same depth as input image)
Optional scale factor	alpha = 1.55
Optional delta added to the scaled values	beta = -65.5

6.2.2 Binarizzazione soglia globale

Parametro	Valore
Global Threshold	80.0