

Waste Material Classification

Erica Bertuzzi, Luca Cotugno, Andrea Mancini

13 febbraio 2024

1 Introduzione

Nella vita di tutti i giorni, accade spesso di dover affrontare la gestione di rifiuti di varia natura, una sfida che richiede una consapevolezza crescente in merito ai materiali che costituiscono gli scarti che generiamo.

Annualmente, infatti, secondo "The World Bank" vengono prodotti 2.01 miliardi di tonnellate di rifiuti urbani, di cui il 33% non smaltito correttamente. La produzione media giornaliera per persona si aggira attorno a 0.76 kg, ma può variare da 0.11 a 4.54 kg. I paesi maggiormente sviluppati, anche se rappresentano soltanto il 16% della popolazione mondiale, producono il 34% dei rifiuti globali.

Gli oltre 2 miliardi di tonnellate di rifiuti annui del 2016, si stima potrebbero salire a oltre 2 miliardi e mezzo nel 2030 e successivamente a 3,4 miliardi entro il 2050.[\[KWW18\]](#)

Inoltre stando al Global Waste Management Outlook del 2015, circa il 24% dei rifiuti sono rifiuti solidi urbani prodotti dai cittadini (a confronto del 21% che sono di natura industriale).

Il riciclaggio può trasformarsi in fonte di guadagno e risparmio, ma affinché ciò sia possibile è essenziale differenziare i rifiuti correttamente.

Non tutti i paesi chiedono alle famiglie di dividere i rifiuti nelle medesime categorie. A Cambridge, per esempio, rifiuti cartacei e plastica vanno nello stesso bidone; a Bologna, invece, carta e plastica vanno separate.

1.1 Descrizione del Problema

Identificare con precisione la composizione di un rifiuto diventa quindi un'operazione ricorrente, e talvolta, un compito impegnativo. Riconoscere le differenze tra i materiali, ad esempio distinguendo una carta plastificata da un'etichetta in plastica, non è sempre immediato e comprendere la biodegradabilità di un determinato rifiuto può essere tutt'altro che banale. Le applicazioni di supporto già esistenti, come "il Rifiutologo" o "Junker", sono capaci d'identificare la categoria d'appartenenza del rifiuto, richiedendo la scansione del codice a barre del prodotto, ma corrono il rischio di non essere in grado di riconoscere determinati articoli, i quali potrebbero non essere presenti nei loro archivi. Esiste inoltre la possibilità che il codice a barre non si trovi sul prodotto da differenziare, ma su di un involucro che è stato rimosso e buttato precedentemente. Inoltre spesso accade che per una sbagliata valutazione, per disattenzione oppure per indifferenza, i rifiuti di un certo materiale vengano inseriti nel bidone sbagliato.

Questi tipi di problematiche, in particolare la prima descritta, possono accadere ad ogni persona sia in ambito domestico sia al di fuori della propria abitazione. La seconda soprattutto a chi si occupa della gestione e programmazione della raccolta dei rifiuti, per esempio un comune o un'azienda.

Nel cercare possibili soluzioni o approcci per migliorare queste situazioni, l'intelligenza artificiale può diventare un valido strumento. Infatti, come mai prima ad ora, il suo impiego sta crescendo a ritmi vertiginosi ed adottata in contesti differenti. Numerosi studi hanno discusso di possibili impieghi dell'intelligenza artificiale nella gestione dei rifiuti, nella logistica del loro deposito e nel loro trasporto. [\[Fan23\]](#) [\[TASG19\]](#) Un esempio è Hooly [\[Gan24\]](#) presentato recentemente al CES2024 di Las Vegas, è un cestino intelligente che tramite un sensore e una telecamera riconosce il tipo di rifiuto e lo smista al suo interno in un contenitore apposito. Questo grazie all'utilizzo di modelli Deep Learning che sfruttano tecniche di Computer Vision per la classificazione. Questo progetto vuole unirsi agli approcci già esistenti per una soluzione che possa migliorare l'esperienza delle persone nella gestione dei rifiuti domestici, oltre ad un possibile impiego da parte delle aziende che si occupano della raccolta dei

rifiuti. Attraverso l'utilizzo di modelli di reti neurali, è possibile classificare il materiale di un rifiuto attraverso un'immagine. Con questo approccio si può ottenere una maggiore efficienza ed attenzione nella suddivisione dei rifiuti per migliorare e velocizzare il processo di riciclaggio. Infatti le aziende di smaltimento potrebbero ridurre il tempo che viene adoperato per controllare la corretta suddivisione dei rifiuti da parte delle persone. Questo comporterebbe una riciclo dei materiale più efficacie e rapido. Questo grazie alle persone che possono avvalersi di uno strumento che li aiuti e guidi nel capire il materiale del rifiuto.

1.2 Soluzione Proposta

Come prima cosa abbiamo cercato di capire quali fossero i materiali riciclabili che maggiormente vengono raccolti in ambito domestico. Generalmente si distinguono tra carta, plastica, vetro e metallo (alluminio ed acciaio). In seguito ci siamo preoccupati di trovare un dataset sufficientemente grande e suddiviso per questi materiali. Per esempio il modello TrashNet proposto in [YT16] è stato allenato con un dataset creato da due studenti di Stanford per un loro progetto. Il dataset è composto da 2527 immagini suddiviso in 6 classi: "glass", "paper", "cardboard", "metal", "plastic" e "trash". Le immagini, di dimensione 512x384, sono state acquisite posizionando gli oggetti su un piano bianco ed illuminato dalla luce del sole o della stanza. Abbiamo deciso poi di affidarci ad architetture neurali, ed in particolare "Convolutional Neural Network CNN", in quanto in questo momento sono lo stato dell'arte nell'apprendimento delle immagini.[KSH12] [LZY⁺17] Infatti grazie alla loro struttura, ogni livello o layer, può imparare a focalizzarsi su aspetti diversi delle immagini. Combinando e comprendendo queste informazioni è possibile riuscire ad imparare una funzione che, per esempio, riconosca e classifichi le immagini. Rispetto ad altri modelli più tradizionali nel Machine Learning, le reti neurali sono più performanti e discriminativi, in più scalabili anche su grandi quantità di dati. Questo aspetto è fondamentale per il nostro obiettivo: riconoscere e classificare i materiali. Una caratteristica fondamentale delle reti neurali è il Transfer Learning. Poder riutilizzare la conoscenza acquisita da un modello nel discriminare ed estrapolare le features più importanti delle immagini è molto utile ed efficace quando, per esempio, non disponiamo di un dataset sufficientemente ampio. Proprio per questo motivo abbiamo utilizzato la parte convoluzionale di una MobileNetV2 addestrata su ImageNet come "backbone" del nostro classificatore.

In primo luogo abbiamo subito notato che il dataset fosse fortemente sbilanciato in particolare verso la classe "trash". Infatti, comparandola con la classe maggiormente rappresentata, "paper", questa è 4 volte superiore in numero, e ispezionando le sue immagini abbiamo notato come queste condividessero molte caratteristiche con le altre classi, includendo sia oggetti di plastica, che di carta e così via. A ciò va aggiunto il fatto che tale classe non sembrava così fondamentale per il nostro problema quanto le altre. Abbiamo pensato quindi che, rimuovendola, si sarebbero potute migliorare le performance del modello. E così è stato. Infatti queste immagini pensiamo che andassero a ridurre e complicare le capacità del modello nel riuscire ad imparare a discriminare le features caratteristiche di ogni classe, o materiale in questo caso.

Considerando che rifiuti di uno stesso materiale possono avere numerose forme e dimensioni diverse è importante che il nostro modello sia efficace nel generalizzare ogni classe. Questo purtroppo non è possibile utilizzando soltanto questo dataset non essendo sufficientemente ampio e variegato. Abbiamo così pensato ad un diverso tipo di approccio per addestrare la nostra rete neurale. In più avendo trovato altri dataset, più ampi come [dat21], ma etichettati in maniera differente a TrashNet abbiamo deciso di spostare il problema da "Supervised" a "Semi-Supervised Learning". In particolare abbiamo utilizzato la tecnica del "Self-Supervised Learning" attraverso Pseudo-Labeling di dati non etichettati, ovvero considerando il dataset [dat21] come non etichettato. Questo processo iterativo ha lo scopo di migliorare il modello allenato con il dataset di TrashNet, arricchendolo con nuovi dati. Questi ultimi vengono etichettati artificialmente dal modello, infatti viene assegnata la label con confidenza maggiore o uguale ad un valore di threshold fissato. Questo metodo può essere applicato in realtà differenti in quanto permetterebbe una personalizzazione del modello in base alla realtà che lo adotta. Infatti partendo da un dataset etichettato di base come TrashNet potrebbe essere possibile arricchire il modello con immagini non supervisionato per specializzarlo ad certo contesto. Per esempio se esistono dei rifiuti particolari di una certa zona con forme e caratteristiche simili ma non identiche a TrashNet, unire immagini non etichettate di questi con TrashNet perfezionerebbe il modello, raggiungendo migliori risultati per l'utilizzo in quella zona.

Il modello finale, allenato tramite Pseudo-Labeling, è stato testato con due test set. Il primo è lo stesso usato in trashnet, il secondo aggiunge al primo immagini provenienti dal dataset che consideriamo non etichettato [dat21], per verificare le performance del metodo pseudo-labeling. L'accuratezza raggiunta in entrambi i test set è dell'80%, confrontando con il valore ottenuto in TrashNet[YT16] abbiamo un miglioramento del 13%. Il confronto principale è stato però fatto tra il modello 'baseline' e il modello allenato con Pseudo-Labeling. In questo caso il 'baseline' raggiunge una accuratezza sul primo test set del 82%, quindi superiore di 2 punti percentuali rispetto al modello finale. Al contrario sul secondo test set più ampio diminuisce a 77%. Possiamo dedurre che il modello finale perde conoscenza su features specifiche delle immagini di TrashNet, a fronte di una migliore capacità di generalizzazione in contesto più ampio come nel secondo test set. Ad ogni modo ci saremmo aspettati miglioramenti più significativi e dei quali discutiamo nelle Conclusioni a paragrafo 4.

1.3 Confronto letteratura

Negli ultimi anni molti lavori sono stati portati avanti per migliorare l'impatto della raccolta differenziata. Molte tecniche basate sull'utilizzo di tecnologie come machine learning o reti neurali sono state utilizzate per automatizzare il riconoscimento dei rifiuti.

Per esempio nel paper [YT16] sono state messe a confronto Support Vector Machine (SVM) con scale-invariant feature transform (SIFT) [Low04] e una Convolutional Neural Network (CNN) per classificare rifiuti. E dai risultati ottenuti si può notare come SVM ha prestazioni nettamente migliori rispetto alla CNN poichè SVM ha un'accuracy del 63% mentre CNN ha un'accuracy solo del 22%.

Arebay et al. [AHBB12] hanno presentato un lavoro che però propone un altro metodo per migliorare la gestione dei rifiuti. Il loro progetto utilizzando una grey level co-occurrence matrix (GLCM) in combinazione con alcune tecnologie di comunicazione come GIS, RFID, GPRS si pone l'obiettivo di monitorare il livello di spazzatura per ogni bidone in modo da ottimizzare la raccolta dei rifiuti e diminuire il numero di mezzi utilizzati per tale scopo. All'interno del paper vengono utilizzati e testati due algoritmi ovvero un Multilayer Perceptron (MLP) e un K-nearest neighbour (KNN) e dai risultati ottenuti il KNN ha performance migliori rispetto ad MLP.

Un'altro importante lavoro [CBP⁺18] è quello proposto dai creatori di Recycle-Net, dove hanno testato diversi modelli di reti neurali e alcuni algoritmi di classificazione per poter dimostrare che le reti neurali sono realmente utilizzabili per un utilizzo del genere. I modelli scelti sono stati VGG-16 e AlexNet mentre gli algoritmi sono K-Nearest Neighbor (KNN), Support Vector Machine (SVM) e Random Forest (RF). Attraverso i risultati ottenuti possiamo notare come i due modelli sono quelli che hanno un accuracy migliore. Più nello specifico abbiamo che VGG-16 ottiene un'accuracy del 93% e AlexNet 91% mentre con KNN, RF e SVM otteniamo rispettivamente un accuracy di 88%, 85% e 80% .

Soprattutto nell'ultimo periodo l'intelligenza artificiale e soprattutto la Computer Vision viene sempre più utilizzata per poter provare a risolvere il problema della raccolta differenziata e gestione dei rifiuti. Ci sono varie soluzioni proposte e molte sono non solo applicabili ad un contesto industriale come si poteva avere qualche anno fa ma adesso queste tecnologie sono facilmente utilizzabili.

I risultati di questi test sono naturalmente influenzabili dal dataset scelto per allenare e testare questi modelli per questo il focus del nostro lavoro non è semplicemente quello di riconoscere il tipo di rifiuti, ma soprattutto utilizzarlo per poter sviluppare un modello in grado di etichettare grandi quantità di immagini non etichettate e richiedendo solo una piccola parte di immagini già etichettate. Poichè come sappiamo etichettare le immagini, testi o qualsiasi altro tipo di dato richiede grande lavoro da parte dell'uomo ed è una delle sfide che si prova a risolvere all'interno del campo dell'intelligenza artificiale. Uno dei metodi proposti è lo Pseudo-labeling, per esempio quello proposto nel paper di FixMatch [SBL⁺20] il quale però non viene proposto in un contesto di raccolta dei rifiuti, ma noi abbiamo adattato al nostro caso d'uso.

Inizialmente abbiamo cercato di suddividerci i compiti in 3 parti, Erica la parte di ricerca dei dati, Luca e Andrea lo studio e ricerca dei metodi esistenti in letteratura. Passati all'implementazione abbiamo cercato di mantenere le stesse divisioni, ma chiaramente nel tempo tutti i componenti hanno implementato varie parti del codice in base alle necessità. Principalmente Andrea ha lavorato sulla

parte di implementazione dell'algoritmo, Luca nella parte di fine-tuning degli iperparametri e ricerca del modello migliore, Erica nella prima parte di preparazione dei dati e tecniche di augmentation. Infine per la parte di evaluation abbiamo tutti contribuito allo stesso modo.

2 Metodo Proposto

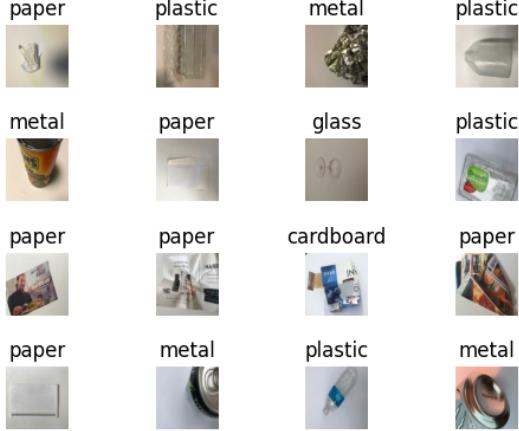


Figura 1: TrashNet's images example

Il primo step è il download dei dataset. Ne abbiamo scelti due diversi entrambi etichettati e divisi quindi in 5 possibili classi che sono: "paper", "cardboard", "plastic", "glass" e "metal", un esempio è riportato in figura I due dataset scaricati da Kaggle avranno utilizzzi differenti. Il primo dataset verrà utilizzato come dataset etichettato per allenare il nostro classificatore, mentre il secondo dataset supponiamo che non sia etichettato e quindi lo utilizzeremo per lo Pseudo-labeling. Da questi due dataset andremo ad ottenere i nostri train set, test set, validation set e unlabeled set. Nello specifico il primo dataset verrà spartito in train set, test set e validation set. Mentre il secondo verrà spartito in test set e unlabeled set. Infine, due test set verranno uniti per ottenere il test set finale. In figura 2 riportiamo la loro composizione. Dopo il download dei dataset andiamo a definire il nostro modello.

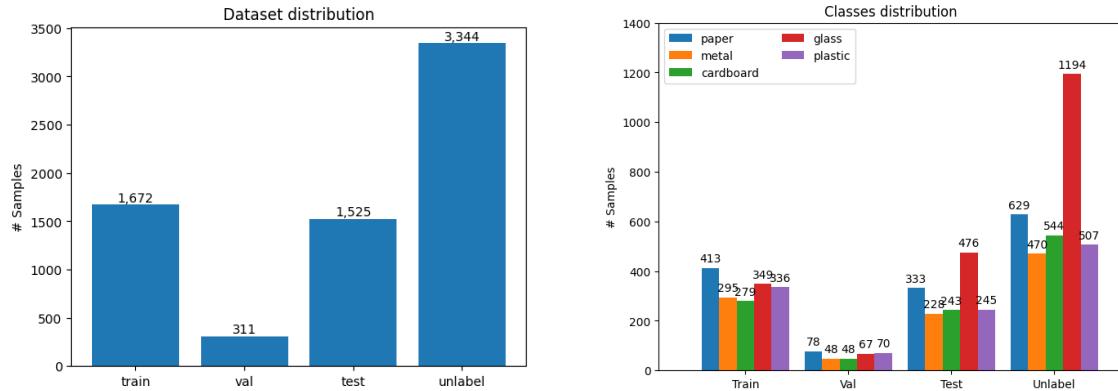


Figura 2: Distribuzione dei dati

Nel nostro caso il modello è composto da una backbone ovvero MobileNetV2, preallenata con i pesi di imagenet e freezata quindi non andiamo a modificarne i pesi durante il training, a cui concateniamo un classificatore finale.

La scelta è ricaduta su MobileNetV2 per via della sua dimensione ridotta e quindi perfetta per un utilizzo su dispositivi mobile cioè il nostro caso d'uso.

Una volta ottenuto il modello andiamo ad allenare il nostro classificatore con il dataset TrashNet scaricato in precedenza.

Prima di passare allo pseudo-labeling è stato testato l'utilizzo del fine tuning sul modello ovvero andare ad unfreezeare alcuni livelli e allenarli con il nuovo dataset. Eseguiti i test però i risultati non portavano migliorie quindi nel metodo finale è stato scelto di non applicare il fine tuning.

Ottenuto il modello con il classificatore allenato adesso passiamo allo pseudo-labeling. Per quanto riguarda il processo di pseudo-labeling abbiamo preso spunto da quello applicato in FixMatch[[SBL+20](#)] riassunto in figura 3.

FixMatch è basato sulla combinazione di due componenti importanti degli algoritmi di Self Supervised

Algorithm 1 FixMatch algorithm.

- 1: **Input:** Labeled batch $\mathcal{X} = \{(x_b, p_b) : b \in (1, \dots, B)\}$, unlabeled batch $\mathcal{U} = \{u_b : b \in (1, \dots, \mu B)\}$, confidence threshold τ , unlabeled data ratio μ , unlabeled loss weight λ_u .
 - 2: $\ell_s = \frac{1}{B} \sum_{b=1}^B H(p_b, \alpha(x_b))$ {Cross-entropy loss for labeled data}
 - 3: **for** $b = 1$ **to** μB **do**
 - 4: $q_b = p_m(y | \alpha(u_b); \theta)$ {Compute prediction after applying weak data augmentation of u_b }
 - 5: **end for**
 - 6: $\ell_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}\{\max(q_b) > \tau\} H(\arg \max(q_b), p_m(y | \mathcal{A}(u_b)))$ {Cross-entropy loss with pseudo-label and confidence for unlabeled data}
 - 7: **return** $\ell_s + \lambda_u \ell_u$
-

Figura 3: Algoritmo FixMatch

Learning: Consistency Regularization e Pseudo-Labeling.

Consistency Regularization si basa sull'idea che un modello dovrebbe restituire in output una classificazione simile quando riceve in input la stessa immagine ma leggermente perturbata. Pseudo-Labeling invece si basa sull'idea di usare il classificatore stesso per ottenere una label artificiale relativa ad un dato unlabeled andando a prendere l'argmax dell'output del classificatore solo però nel caso questo superi il valore di un dato threshold.

Per quanto riguarda la consistency regularization all'interno dell'algoritmo verranno applicate alle immagini unlabeled due tipi di image augmentation che chiameremo rispettivamente weak e strong augmentation, riportiamo un esempio in figura 4. Nella weak augmentation l'immagine sarà flippata orizzontalmente e verticalmente e verranno applicati randomicamente del Gaussian Blur e una rotazione all'immagine. Nella strong invece all'immagine sarà applicato un Dropout di pixel e randomicamente verranno applicate delle trasformazioni affini che andranno a scalare, roteare, translare e applicare un taglio all'immagine.



Figura 4: Augmentation Example

La funzione di loss dell'algoritmo è formata da due cross-entropy loss ovvero una supervised loss ottenuta dai sample labeled e un unsupervised loss ottenuta dai valori unlabeled. Per la supervised loss abbiamo semplicemente una categorical cross entropy tra la label reale e la label restituita dal modello. Mentre l'unsupervised loss è ottenuta da una categorical cross entropy tra l'output del modello relativo all'immagine con weak augmentation e l'output del modello relativo all'immagine con strong

augmentation. Come già detto in precedenza verranno prese in considerazione solo i valori in output che superano un certo threshold.

Una volta che viene applicata l'augmentation alle immagini, ottenute le loro predizioni e calcolata la function loss vengono aggiornati i pesi del modello e le immagini che hanno ottenuto una pseudolabel che appunto supera il threshold vengono aggiunte al train set proprio con la label predetta. Dopodichè il processo viene ripetuto dall'inizio in modo che il modello riesca a predire sempre più immagini sopra la soglia di confidenza scelta.

2.1 Scelta della Soluzione

Per misurare le performance del nostro modello utilizziamo due dataset: validation e test set. Il set di validation è composto da immagini solamente provenienti dal dataset di TrashNet, questo è utile per catturare anche la differenze di performance tra i due modelli. Il set di test invece è composto da immagini di entrambi i dataset, importante per evidenziare la differenze di prestazioni con l'utilizzo di tecniche di self-training come FixMatch per arricchire la conoscenza e generalizzazione del modello con immagini non etichettate. Infine abbiamo scelto di concentrarci principalmente sul trade-off tra le metriche 'precision' e 'recall'. In particolare siamo interessati alla 'precision', ovvero la proporzione dei rifiuti effettivamente classificati correttamente.

Per questo abbiamo scelto di mostrare la 'precision recall curve', molto utile per indicarci le performance del nostro modello rispetto appunto al trade-off sopracitato. E la confusion matrix per capire quali sono i tipi di materiale che più difficilmente vengono classificati correttamente, e per quali erroneamente (false positive).

3 Risultati Sperimentali

3.1 Dimostrazione e Tecnologie

Come ambiente di lavoro abbiamo scelto Colab, sviluppando quindi un Jupyter Notebook. Questo soluzione facilità molto l'eseguibilità del codice e la riproducibilità. Come precedentemente descritto i due dataset utilizzati vengono scaricati dal framework Kaggle. Inserendo quindi il percorso del proprio token è possibile eseguire poi tutte le celle seguenti. Abbiamo scelto di utilizzare Tensorflow come framework per l'implementazione, principalmente perché tutti i componenti del gruppo hanno più confidenza con esso. Per la fase di valutazione del modello abbiamo sfruttato le librerie python di 'scikit' e 'matplotlib' per calcolare le metriche e mostrare i relativi grafici.

3.2 Risultati

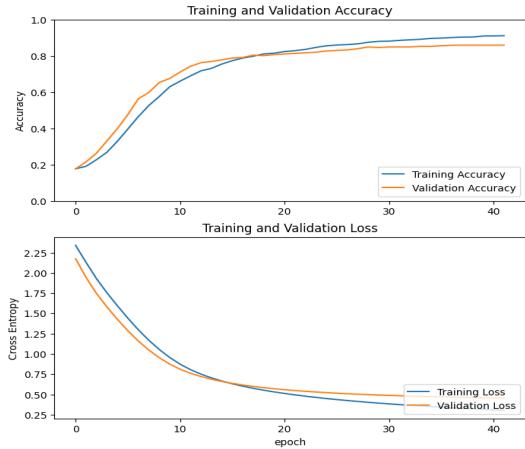
Nella ricerca della configurazione migliore del modello, abbiamo ripreso alcuni parametri già studiati in [FixMatch] come Exponential Moving Average con momentum = 0.999, per regolare durante la discesa del gradiente i dati rumorosi ed evidenziare i pattern più significativi. Abbiamo riportato in tabella i parametri che influiscono di più nella performance del modello: learning rate e weight decay. Riguardo al primo abbiamo utilizzato un Cosine Decay Scheduler con warmup sulle prime 10 epoch. Non abbiamo riportato nella tabella il 'batch size' perché abbiamo notato che questo non influisse nelle prestazioni.

Ricordiamo che abbiamo scelto di misurare le performance dei due nostri modelli utilizzando due dataset di test. Il primo è il medesimo utilizzato in TrashNet, mentre il secondo aggiunge immagini provenienti dal secondo dataset, di cui il resto consideriamo non etichettato.

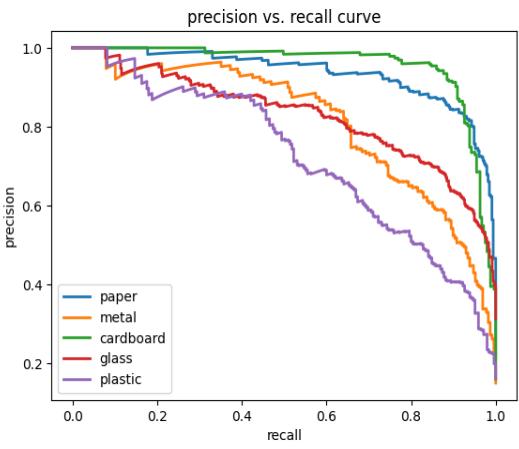
Entrambi i modelli sono stati allenati utilizzando Adam come SGD optimizer. Per cercare di rimediare al problema di overfitting, in particolare nel primo modello, utilizziamo Early Stopping con patience a 5. Alleniamo infine per un massimo di 100 epoche, ma in nessun caso sono state mai raggiunte.

Il modello baseline, allenato soltanto sul dataset TrashNet, utilizza Cosine Decay con learning rate iniziale a 1e-4, weight decay a 5e-6 ma senza EMA. Abbiamo visto che EMA riduce di un fattore decimale, 0.1, precision e recall. Raggiungiamo un'accuratezza sul primo test set del 82%, sul secondo 77%.

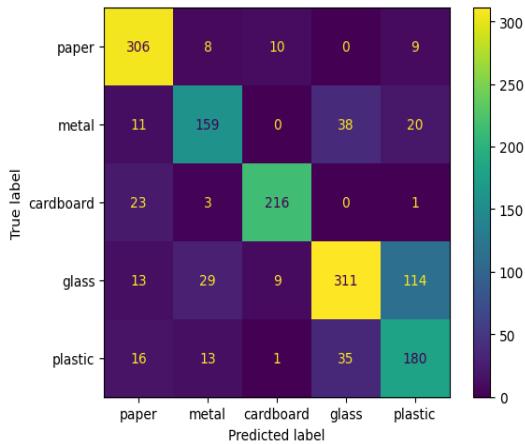
Ci siamo concentrati principalmente a studiare il secondo modello allenato attraverso l'algoritmo FixMatch, utilizzando quindi sia dataset etichettati dal dataset TrashNet che dataset non etichettati



(a) Training Metrics Baseline



(b) Precision-Recall Curve Baseline



(c) Confusion Matrix Baseline

da [dat21]. Come il primo modello, utilizziamo come ottimizzatore Adam ma con EMA e momentum a 0.999, come riportato in FixMatch. Applichiamo sempre Early stopping e Cosine Decay Scheduler per regolare l’overfitting. Abbiamo ricercato i valori ottimali di learning rate, weight decay ed infine Threshold.

Al variare di quest’ultimo, infatti, dipendono in modo particolare le performance del modello. Infatti la quantità delle immagini etichettate tramite pseudo-labelling è inversamente proporzionale al valore del threshold, al contrario della precisione delle predizioni.

Come prima cosa abbiamo allenato il modello con diversi valori di learning rate iniziale e weight decay, con valore fisso del threshold a 0.9, valore medio di quelli presi in considerazione. I risultati sono in riportati in tabella 1.

Scelti i valori di learning rate e weight decay con accuratezza migliore su validation set, investighiamo le performance del modello a diversi valori del Threshold. Il grafico 6 mette in relazione i valori di ‘precision’ e ‘recall’ compresi tra 0.8 e 0.99, calcolati sul test set.

Osserviamo che con valore di threshold a 0.825 abbiamo il valore di precision più alto, 0.803. In questo modo abbiamo scelto il modello finale con performance migliori in termini di precisione. Riepilogando abbiamo come learning rate iniziale $1e-4$, weight decay $5e-6$ e threshold 0.825. Come per il primo modello mostriamo i medesimi grafici di valutazione sul test set.

Confrontando già il nostro primo modello che sfrutta il Transfer Learning grazie all’utilizzo di MobileNetV2 come feature extractor, abbiamo risultati superiori al modello SVM di TrashNet [YT16], 0.82% rispetto a 0.63% su medesimo test set. Lo stesso viene riportato confrontando ‘precision’ e ‘recall’ in tabella 2.

Rispetto invece al nostro modello migliore allenato con FixMatch [SBL+20], riusciamo ad avere una accuratezza maggiore sul secondo test set, 80% contro un 77%.

l-rate	weight decay	val accuracy
5e-4	5e-3	0.826367
5e-4	5e-4	0.842444
5e-4	5e-5	0.852090
5e-4	5e-6	0.845659
1e-4	5e-3	0.836013
1e-4	5e-4	0.836013
1e-4	5e-5	0.832797
1e-4	5e-6	0.858521
5e-5	5e-3	0.794212
5e-5	5e-4	0.813505
5e-5	5e-5	0.816720
5e-5	5e-6	0.784566

Tabella 1: Learning Rate and Weight Decay study

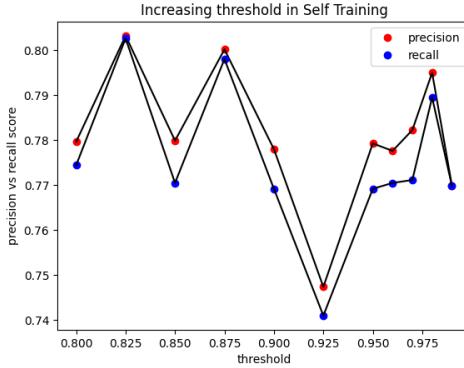


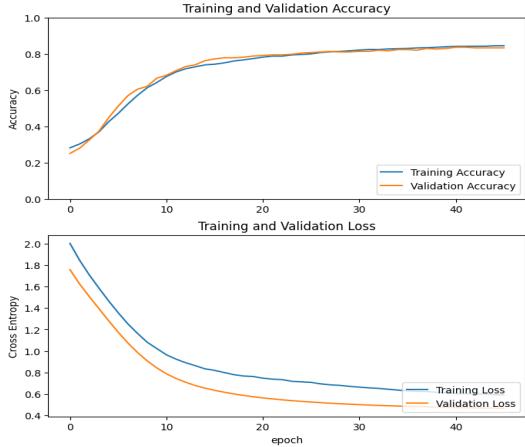
Figura 6: Threshhold Comparison

material	SVM precision	Our precision	SVM recall	Our recall
glass	0.55	0.81	0.60	0.65
paper	0.80	0.83	0.70	0.92
cardboard	0.62	0.92	0.66	0.89
plastic	0.61	0.56	0.69	0.73
metal	0.70	0.75	0.70	0.70

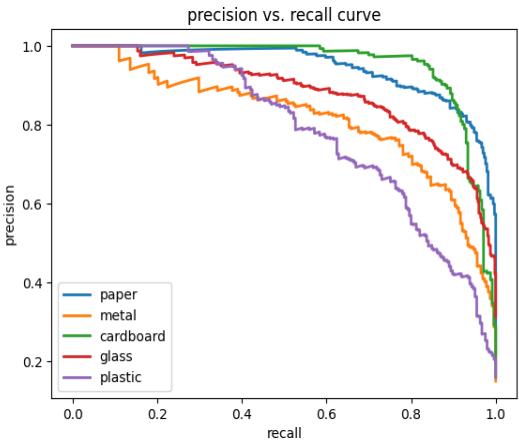
Tabella 2: Confronto Precision Recall con SVM TrashNet

L'utilizzo di vari tipi di trasformazione applicati in maniera 'weak' e 'strong' sui dataset non etichettati aiutano a migliorare il problema di overfitting rispetto ad altri metodi utilizzati in precedenza. In particolare come l'utilizzo della trasformazione Dropout influenzò positivamente il comportamento del modello in fase di training prevenendo overfitting.

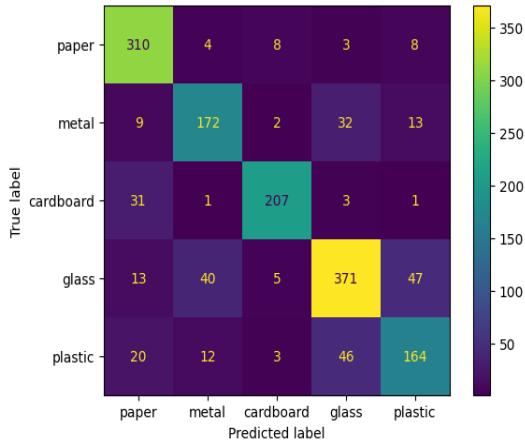
Una comparazione con altri modelli in letteratura è possibile farla con WasteNet[GW20] per esempio. Similmente a noi, utilizza una DenseNet pre-addestrata su ImageNet come backbone del modello e come dataset TrashNet. La particolarità della loro soluzione è l'utilizzo di una tecnica di fine-tuning del modello ibrido, ovvero i layer vengono scongelati gradualmente al crescere dell'accuratezza e allo stabilizzarsi del loss. Questo gli ha permesso di raggiungere un'accuratezza del 97% sul test set, lo stesso del nostro primo, stessa percentuale per precision e recall. In questo caso la comparazione è più corretta farla con il nostro modello 'baseline' che è stato addestrato in 'Supervised' e solamente con il dataset di TrashNet come WasteNet[GW20]. Presumiamo quindi che DenseNet si potrebbe adattare meglio a questa tipologia di immagini, sicuramente in lavori futuri sarebbe interessante testare questa soluzione coniugando fine-tuning ibrido e FixMatch.



(a) Training Metrics FixMatch



(b) Precision-Recall Curve FixMatch



(c) Confusion Matrix FixMatch

Eravamo anche curiosi di capire quali fossero le parti delle immagini che contribuissero di più nella decisione del classificatore. Abbiamo utilizzato come tecnica GradCam[RR19]; questa localizza, attraverso l'applicazione dei gradienti calcolati dalla classificazione sulla feature map ottenuta dal convolutional layer finale, una zona di heatmap nell'immagine, ovvero la zona focale in cui si è concentrato il modello per la classificazione.

In figura 8 abbiamo testato GradCam con una immagine semplice di una bottiglia di plastica sopra ad un tavolo. Il modello ha predetto correttamente il materiale. Da notare che questa differisce anche dalla tipologia di immagini con cui il modello si è allenato, ovvero ben illuminata e su sfondo bianco. Possiamo vedere come il modello riesca a riconoscere la forma dell'oggetto.

4 Discussione e Conclusioni

Confrontando i due nostri modelli ci saremmo aspettati differenze più marcate utilizzando l'algoritmo FixMatch. Inizialmente ci saremmo aspettati che utilizzando più immagini durante l'addestramento le prestazioni sarebbero dovute aumentare in quanto queste avrebbero dovuto aiutare il modello a discriminare meglio tra i vari materiali e quindi generalizzare più efficacemente.

La minore accuratezza sul primo test set del modello finale è possibile aspettarsela in quanto il modello perde conoscenza di features specifiche e caratteristiche delle immagini di TrashNet. Queste vengono sostituite in fase di Pseudo-Labeling da informazioni apprese dalle immagini non etichettate, le quali aiutano il modello ad estendere le sue performance di classificazione in un contesto più generale come con il secondo dataset di test. Infatti con FixMatch andiamo ad introdurre molte nuove immagini che differiscono sicuramente con TrashNet.

Sicuramente questo metodo è molto utile se si hanno a disposizione una quantità molto limitata di

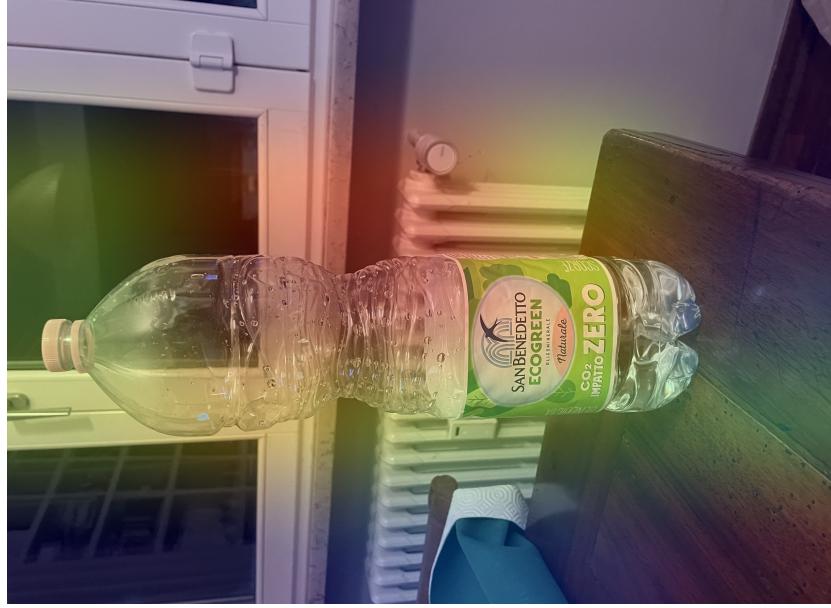


Figura 8: GradCam test

immagini etichettate e allo stesso tempo un grande numero di immagini non etichettate. Infatti un limite importante da tenere in conto in fase di training, e presente nel nostro specifico caso, è il numero di immagini etichettate con pseudo label ad ogni epoca. Abbiamo notato che all’incirca tra due epoche successive l’incremento di immagini è di circa 20-40. In più nel nostro caso molte immagini non etichettate non vengono mai considerate, in quanto il modello prende in considerazione soltanto quelle che superano il threshold fissato. Se è vero che ad ogni step le prestazioni del modello migliorano, queste non sono sufficienti ad arricchire in modo significativo allo step successivo il numero di immagine pseudo etichettate. All’incirca più di 1000 immagini non vengono mai utilizzate, molto probabilmente perché gli oggetti sono di forme e caratteristiche troppo diverse rispetto a quelle con cui si è allenato fino a quel momento anche se dello stesso materiale. Per questo pensiamo che avere un dataset non etichettato dell’ordine di decine di migliaia di immagini possa fare davvero la differenza nel nostro caso.

Fondamentale e limite del metodo è scegliere il giusto valore del threshold. Nel nostro caso per esempio avendo un numero non considerevole di immagini non etichettate, notiamo dai risultati, fig. 6, che il modello ha risultati migliori con valore 0.825 che è decisamente inferiore a quello consigliato da FixMatch[SBL⁺20], ovvero 0.95. Supponiamo che con pochi dati è più importante per il modello alleinarsi con più immagini ma con precisione inferiore, rispetto a pochissime immagini ma con precisione maggiore. D’altro canto sempre nello stesso grafico 6, non è ben delineato il comportamento al variare del threshold. Possiamo notare infatti fluttuazioni che non mostrano un pattern preciso. Notiamo però una leggera tendenza decrescente del valore di ‘precision’. Lo stesso comportamento lo ritroviamo anche con il valore di ‘recall’, ma è quello che ci possiamo aspettare all’incremento del threshold.

Registriamo nonostante tutto un miglioramento tra i due modelli infatti se consideriamo le rispettive Confusion Matrix, 5c e 7c, possiamo notare come il numero di elementi di vetro classificati come plastica si riduce di più della metà. Infatti la misclassificazione tra oggetti di vetro e plastica è uno dei bias più importanti e difficili anche da gestire. Sono tanti gli oggetti fatti con questi due materiali che condividono la stessa forma, stesso colore e caratteristiche.

4.1 Limitazioni e Maturità

Consideriamo il nostro lavoro ad una fase ancora iniziale, non utilizzabile in un contesto professionale e nemmeno domestico. Infatti con questo metodo, ma anche più in generale, crediamo che sia fondamentale avere a disposizione grandi quantità di dati. Questo per aumentare la qualità di generalizzazione del modello, che in un caso d’uso come il riconoscimento di rifiuti è estremamente importante. Questo è dovuto dal fatto che esistono moltissimi tipi di rifiuti, e quelli che condividono lo stesso materiale

allo stesso tempo possono avere forme, pattern e dimensioni totalmente diverse. Per questo motivo crediamo che l'approccio self-training non si presti particolarmente bene a questo tipo di problema, in quanto è una tecnica molto utile nel caso in cui si hanno poche immagini etichettate. Nel nostro caso è necessario avere per ogni materiale avere tante immagini etichettate che comprendono più oggetti e forme possibili. Questo è dovuto dal fatto che il modello si concentra sulle feature visive, come appunto la forma, gli angoli ed il colore. Sarebbe più significativo per il modello allenarsi su features più specifiche legate al materiale del rifiuto.

4.2 Lavori Futuri

Possibili lavori futuri possono incentrarsi sul raccogliere un numero maggiore di rifiuti domestici per migliorare le prestazioni dell'algoritmo FixMatch. Investigare più a fondo sulle possibili trasformazioni da applicare sui dati non etichettate, in particolare quelle definite come 'strong', che sono presentate e testate su FixMatch [SBL⁺20]. Un altro possibile approccio sarebbe poter utilizzare telecamere che possano risaltare le caratteristiche tipiche di ogni materiale, per esempio che risaltino la differenza nei vari tipi di riflesso superficiale o che ne evidenzino pattern caratteristici. Inoltre provare ad utilizzare altri modelli convoluzionali più performanti e attuali di MobileNetV2, per esempio applicare il metodo descritto precedentemente in WasteNet[GW20] con FixMatch. Inoltre sperimentare l'uso di reti come YOLO per object detection nel caso in cui abbiamo più rifiuti in una stessa immagine.

Riferimenti bibliografici

- [AHBB12] Maher Arebey, M.A. Hannan, R.A. Begum, and Hassan Basri. Solid waste bin level detection using gray level co-occurrence matrix feature extraction approach. *Journal of Environmental Management*, 104:9–18, 2012.
- [CBP⁺18] Bernardo Costa, Aiko Bernardes, Julia Pereira, Vitoria Zampa, Vitoria Pereira, Guilherme Matos, Eduardo Soares, Claiton Soares, and Alexandre Silva. Artificial intelligence in automated sorting in trash recycling. pages 198–205, 10 2018.
- [dat21] Garbage classification. Kaggle, 2021. <https://www.kaggle.com/datasets/mostafaabla/garbage-classification>.
- [Fan23] Yu Jiacheng Chen Zhonghao Osman Ahmed I. Farghali Mohamed Ihara Ikko Hamza Es-sam H. Rooney David W. Yap Pow-Seng Fang, Bingbing. Artificial intelligence for waste management in smart cities: a review. *Environmental Chemistry Letters*, 08 2023.
- [Gan24] Nicholas Zeoli Ganiga. Hoooly, il cestino intelligente che riconosce e differenzia rifiuti. 2024.
- [GW20] Andrei Palade Fan Li Siobha n Clarke Gary White, Christian Cabrera. Wastenet: Waste classification at the edge for smart bins. 2020.
- [KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012.
- [KVV18] Yao L.C. Bhada-Tata P. Kaza, S. and F. Van Woerden. What a waste 2.0: A global snapshot of solid waste management to 2050. *World Bank*, 2018.
- [Low04] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.
- [LZY⁺17] Qing Liu, Ningyu Zhang, Wenzhu Yang, Wang Sile, Zhenchao Cui, Xiangyang Chen, and Liping Chen. A review of image recognition with deep convolutional neural network. pages 69–80, 07 2017.
- [RR19] Michael Cogswell-Abhishek Das Ramakrishna Vedantam Devi Parikh Dhruv Batra Ramprasaath R, Selvaraju. Grad-cam: Visual explanations from deep networks via gradient-based localization. 2019.

- [SBL⁺20] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *CoRR*, abs/2001.07685, 2020.
- [TASG19] Erfan Babaee Tirkolaee, Parvin Abbasian, Mehdi Soltani, and Seyed Ali Ghaffarian. Developing an applied algorithm for multi-trip vehicle routing problem with time windows in urban waste collection: A case study. *Waste Management & Research*, 37(1_suppl):4–13, 2019. PMID: 30761957.
- [YT16] Mindy Yang and Gary Thung. Classification of trash for recyclability status. *CS229 project report*, 2016(1):3, 2016.