Universitat Autònoma de Barcelona

Dipòsit digital de documents de la UAB

This is the **published version** of the bachelor thesis:

Puga Esquerra, Juan Manuel; Antens, Coen Jacobus, dir. Una aplicació que ajuda a reciclar components de PCBs. 2023. (Enginyeria Informàtica)

This version is available at https://ddd.uab.cat/record/280702

under the terms of the  license

Detecció d'imatges aplicada al reciclatge de components de PCBs

Juan Manuel Puga Esquerra

Resum—La brossa electrònica (coneguda generalment pel terme anglès "*E-Waste*") ha esdevingut en una de les majors preocupacions actuals. Només un 17% d'aquesta brossa és tractada correctament i la resta té efectes molt negatius en termes mediambientals, econòmics, socials i, fins i tot, sanitaris. L'augment continu de la conscienciació social juntament amb una major accessibilitat a les noves tecnologies fan possible l'aparició de propostes que tenen com a objectiu crear solucions a problemes actuals. Dins d'aquestes noves tecnologies, les basades en intel·ligència artificial han experimentat un creixement de popularitat que ha produït l'aparició d'eines molt potents a l'abast de qualsevol.

En aquest projecte, es fa ús d'un model de detecció d'imatges com és YOLO per a detectar els components electrònics que conformen les plaques PCB. Aquesta identificació és una part important en el reciclatge d'aquests components i, generalment, són processos manuals i poc eficients. Un dels objectius principals del present treball és la utilització de les versions més recents d'aquestes tecnologies per tal d'aconseguir optimitzar el reciclatge d'aquesta brossa electrònica i proporcionar una eina d'etiquetatge per a aquests components.

Paraules clau—Models de detecció d'imatge, YOLO 8, PyQt 6, Brossa electrònica, Reciclatge de components electrònics.

Abstract—Electronic waste, commonly referred to as "E-Waste," has emerged as a prominent contemporary issue of utmost significance. Only 17% of this waste is appropriately managed, while the remaining portion exerts profoundly adverse impacts on the environment, economy, society, and even public health. The continuous increase in social awareness, coupled with greater accessibility to new technologies, enables the emergence of proposals aimed at creating solutions to current problems. Within these new technologies, those based on artificial intelligence have experienced a surge in popularity, leading to the emergence of highly potent tools accessible to all.

In this project, a YOLO image detection model is employed to identify the electronic components that constitute PCB boards. This identification is a crucial aspect of recycling these components, and typically, these processes are manual and inefficient. One of the main objectives of this study is to utilize the most recent versions of these technologies in order to optimize the recycling of electronic waste and provide a labeling tool for these components.

Index Terms—Image detections models, YOLO 8, PyQt 6, Electronic waste, Electronic Component Recycling.

1 Introducció - Context del treball

AUTOMATITZAR tasques ha permès fer més eficients processos ja existents i ha suposat també l'aparició de noves formes d'incrementar la productivitat a una gran varietat de sectors. Segons Fernando Silvestre, Country Manager Región Andina en NEORIS, es pot arribar a automatitzar fins al 30% de les tasques a una organització amb una robotització estàtica i fins a un 90% utilitzant intel·ligència artificial. [1]

Un dels camps on s'ha vist incrementat aquest ús de noves tecnologies per tal d'assolir processos més eficients és el tractament de l'anomenat *E-Waste* o "brossa electrònica". Aquest concepte és molt recent i una de les grans preocupacions actualment degut al seu creixement massiu. Segons dades del 2019, s'estima que cada ciutadà europeu genera, de mitja, 16.2 kg d'aquesta brossa anualment i que només el 17.4% es tracta correctament. [2] La importància i la criticitat del *E-Waste* radica en el fet que molts components electrònics contenen metalls perjudicials per al medi ambient. Uns altres factors que promouen cada cop més l'interès pel reciclatge d'aquests

components són la creixent mancança dels materials primers emprats per la seva fabricació i l'augment de la consciència ambiental de la població.



Fig. 1: Foto d'Agbogbloshie (Ghana) proporcionada per Fairphone.

A la Figura 1 podem apreciar un dels fets que més il·lustren el problema que motiva aquest treball. Agbogbloshie és un barri de la ciutat d'Acra (Ghana) que s'ha transformat en una deixalleria per a elements electrònics. Aquesta barriada abasta a prop de 10.5 hectàrees on habiten aproximadament unes 40.000 persones. La situació actual d'aquest barri suposa un perill per a la salut dels residents i del medi a causa de l'exposició contínua a metalls com mercuri, plom o cadmi. Per una altra banda, la presència d'aquesta brossa ha transformat Agbogbloshie en un punt d'interès per a màfies que, fent ús de treball infantil, obtenen i venen aquests valuosos metalls. [3]

Les plaques PCB no són l'excepció a aquest problema i es llencen sense comprovar si els components que les conformen es poden reutilitzar o si es tracten de forma que no esdevinguin un perill per al medi ambient. Als últims anys han sorgit moltes propostes que utilitzen tecnologies com el *Deep Learning* per tal d'aconseguir una detecció i classificació automàtica dels components possiblement reutilitzables (propostes com, per exemple, la de Weinertetal a 2017). [4]

Per a finalitzar, cal esmentar que actualment la majoria dels processos relatius al reciclatge d'aquests components continuen sent manuals i, a causa d'això, poc eficients. La intenció d'aquest treball és desenvolupar un sistema capaç de realitzar la detecció i classificació d'aquests components mitjançant el *Deep Learning* juntament amb xarxes neuronals.

1.1 Objectius

Una vegada determinats tant el context com la intenció del treball, cal esmentar els objectius a complir per assolir un sistema funcional de detecció i classificació de components en plaques PCB. Els objectius es presenten en l'ordre en què s'han d'assolir:

- Disseny i implementació de la interfície gràfica del SW.
- Investigació i decisió del model de *Deep Learning* més adient per al projecte.
- Recol·lecció de material per a conformar els datasets (conjunt d'imatges amb les quals treballarem).
- Realització de proves per confirmar que el sistema detecta i classifica correctament els components.
- Integració del model *Deep Learning* ja entrenat al SW anteriorment implementat.

1.2 Metodologia

Els factors principals que han protagonitzat la decisió de la metodologia a seguir al desenvolupament d'aquest projecte han sigut dos: el temps amb el qual comptem i la dificultat del mateix projecte.

A causa de la complexitat de la tasca a realitzar, es consideren les metodologies anomenades Àgils com les més recomanables gràcies a la seva flexibilitat. Dins d'aquest grup de metodologies, i tenint en compte el temps disponible, s'utilitzarà una variant d'*Extreme Programming* (XP) especialment pensada per treballs individuals (*Extreme Programming For One*). [5] Aquesta variant d'XP manté la majoria d'aspectes essencials eliminant la programació en parelles, però es manté la filosofia original: Implementar la solució més senzilla que pugui funcionar.

Finalment, el control del progrés de les tasques que conformen el projecte es duu a terme mitjançant Jira. Jira és una eina emprada principalment quan la metodologia a seguir és *Scrum* (una altra metodologia àgil). En el nostre cas, no farem servir la programació de *Sprints* (característica principal de *Scrum*) i fem ús únicament del taulell. El taulell de Jira permet enregistrar les tasques i assignar atributs com el seu tipus, una descripció i un estat (per fer, en progrés, finalitzada o error).

2 Estat de l'art

La utilització de tecnologies com la intel·ligència artificial i les xarxes neuronals ha crescut exponencialment a la darrera dècada i cada cop són més els camps que l'implementen. Per parlar de l'estat actual de certes tecnologies, aquesta secció constarà de tres subseccions: interfície gràfica, datasets i models de detecció d'imatges.

Cada part de les esmentades inclou les principals tecnologies que han sigut considerades per a cada aspecte i també els motius d'aquestes eleccions.

2.1 Interfície Gràfica

Una de les funcions d'aquest projecte és permetre l'etiquetatge manual d'imatges tal com ho fan eines existents com LabelImg, Supervisely o LabelBox. Aquesta funcionalitat implica la implementació d'una interfície gràfica.

Les opcions considerades en aquest punt han sigut SimpleGUI, StreamLit i PyQt. A continuació, es llisten els seus punts forts i febles:

- SimpleGUI
  - Els punts forts són la seva facilitat d'ús i que la documentació és extensa.
  - Per una altra banda, tant la compatibilitat i la funcionalitat d'aquesta

llibreria és limitada.

- StreamLit
  - La facilitat d'ús i una comunitat molt activa són les seves principals atraccions.
  - Presenta problemes de seguretat i escalabilitat juntament amb una funcionalitat limitada.
- PyQt
  - Els seus principals aspectes positius són la seva àmplia funcionalitat, un gran rendiment i una documentació completa.
  - Aquesta llibreria és, de les opcions comentades, la que té una corba d'aprenentatge més elevada.

L'elecció final de llibreria per la interfície gràfica ha sigut PyQt. El motiu principal per aquesta decisió és prioritzar la possibilitat d'implementar interfícies més avançades encara que la dificultat sigui major. Cal esmentar que un altre punt important de PyQt és que permet l'execució en diferents plataformes sense necessitat de fer codi específic.

Per a finalitzar aquesta subsecció, cal concretar que la versió específica escollida és PyQt6, ja que presenta millores de rendiment respecte a les versions anteriors d'aquesta mateixa llibreria.

2.2 DATASETS

Un dels efectes de la proliferació de models de detecció i classificació d'imatges és l'aparició de diversos datasets públics a pàgines web com, per exemple, Roboflow. [6] Aquest datasets són conformats majoritàriament per usuaris que fan proves amb aquests models i, a causa d'això, trobar un conjunt de qualitat és complex.

Per finalitzar aquest punt, fer especial menció al projecte "PCB-METAL". [7] Aquest treball fa la seva aparició en 2019 i consta de 984 imatges d'alta resolució de 123 plaques PCB úniques. El dataset considera quatre tipus de components (IC, Capacitadors, Resistències i Inductors) i té com a objectiu facilitar l'entrenament de models *Deep Learning* (encara que l'accés a aquest conjunt és quasi impossible actualment).

2.3 MODELS DE DETECCIÓ D'IMATGES

En els darrers anys, i gràcies a la popularització de la intel·ligència artificial, són molts els algoritmes de detecció d'imatges que han sorgit. Cadascú d'aquests algoritmes té les seves pròpies característiques i àrees d'aplicació especifiques. A continuació, llistarem els principals comentant de manera superficial quins són els seus avantatges i desavantatges:

- SSD (*Single Shot Detector*) → És un algoritme ràpid i fàcil d'entrenar. El seu principal desavantatge és que pot presentar problemes amb formes complexes i no sempre és idoni per tasques molt especifiques. [8]
- FASTER R-CNN → Algoritme d'alta precisió amb gran capacitat per a detectar objectes de diferents mides i complexitats. La contrapart és que és un model lent que fa impossible la seva utilització en aplicacions de detecció en temps real. [9]
- YOLO (*You Only Look Once*) → Model que permet la detecció en temps real amb una precisió raonable. YOLO pot tenir problemes amb objectes petits o de formes complexes. [10]

Després de presentar els principals models que existeixen en l'actualitat, cal parlar de quin és més convenient per l'assoliment de l'objectiu proposat. Una de les funcionalitats del SW a implementar en aquest treball és la possibilitat de detecció en temps real i, a causa d'això, el model Faster R-CNN no és adequat. Per una altra banda, YOLO i SSD han sigut els dos models que més han aparegut a l'hora d'investigar projectes similars al nostre i són les principals opcions a tenir en compte donades les seves característiques.

Començarem parlant del model conegut com a SSD. El funcionament d'aquest model es basa en la utilització de "*Grids*" per a dividir la imatge en petites regions. Cada cel·la d'aquest "*Grid*" és responsable de la detecció d'imatges en la regió que delimita. Aquesta forma de separar la imatge i el fet que SSD realitza una única "passada" sobre la imatge, resulta en un dels models de detecció més ràpids amb una precisió relativament alta. Una vegada indicades les seves propietats favorables, cal indicar aquelles característiques que converteixen al model SSD en una opció menys viable que YOLO en el nostre cas. L'algoritme *Single Shot Detector* pateix a l'hora de detectar objectes petits i la seva velocitat decau significativament quan el model en qüestió és gran (anul·lant així una de les seves principals virtuts).

A continuació, parlarem del model escollit per a aquest treball: YOLO. *You Only Look Once* neix l'any 2016 i ha esdevingut un dels algoritmes més populars de detecció d'imatges. Aquesta popularitat és causada, en gran part, per una comunitat molt activa i una evolució ràpida de l'algoritme (la versió 8 ha sortit set anys després del seu naixement). Les propietats que el fan idoni per al cas que ens ocupa són:

- Permet la detecció en temps real.
- Molt recomanable per a detectar múltiples objectes de múltiples classes alhora.
- Aparició de versions que compensen els problemes a l'hora de tractar amb objectes petits.
- Bona documentació juntament amb una comunitat activa i col·laborativa.

Per a finalitzar aquesta secció, és important indicar que als darrers anys han aparegut moltes propostes que, com aquest treball, busquen l'automatització del reciclatge de components electrònics. Un dels exemples més recents data del gener de 2022 i proposa la utilització de YOLO per a la detecció d'aquests components. [11] Encara que el nombre de propostes cada vegada és major, les aplicacions amb un objectiu similar al nostre utilitzen versions molt antigues del model. Per una altra banda, aquestes propostes fan una separació manual dels components abans de realitzar la detecció. El nostre projecte, però, fa servir YOLO V8 (una versió llançada aquest mateix gener de 2023) i fa la detecció sobre la placa sense dur a terme separació de components.

3 PLANIFICACIÓ INICIAL

Les tasques que conformen la planificació d'aquest treball es poden trobar a la Figura 2. En aquesta taula es poden observar la durada de la tasca en dies juntament amb la

data d'inici i la de final. Cal esmentar que les tasques relacionades amb el Dataset a utilitzar per a l'entrenament del model YOLO depenen de la creació de la interfície gràfica (ja que l'etiquetatge es realitzarà amb l'eina implementada a causa d'aquest treball).

Per una altra banda, el temps necessari per a la conformació d'aquest Dataset engloba la cerca de conjunts públics d'imatges etiquetades (principalment a Roboflow) i la seva ampliació amb les imatges proporcionades per l'ICTA. La importància d'aquestes tasques justifica la inversió de temps, ja que la qualitat del model entrenat depèn de la qualitat del Dataset seleccionat.

A la Figura 2 es presenta la planificació temporal del projecte en forma de diagrama de Gantt. La llista de tasques i el diagrama estan disponibles per a la seva consulta a l'apartat "Apèndix" d'aquest document (A1 i A2 respectivament).

ID	Name	Start Date	End Date	Duration	Progress %	Dependency
1	Interfície Gràfica	Mar 10, 2023	May 03, 2023	39 days	100	
2	Investigació de Llibreries	Mar 10, 2023	Mar 16, 2023	5 days	100	
3	Creació de l'estructura bàsica	Mar 17, 2023	Mar 23, 2023	5 days	100	2FS
4	Finestra d'etiquetatge	Mar 24, 2023	Apr 20, 2023	20 days	100	3FS
5	Finestra d'exportació	Apr 21, 2023	Apr 28, 2023	6 days	100	4FS
6	Refinament del codi	May 01, 2023	May 03, 2023	3 days	100	5FS
7	Dataset	May 04, 2023	May 24, 2023	15 days	100	
8	Cerca de datasets públics	May 04, 2023	May 12, 2023	7 days	100	
9	Etiquetatge d'imatges de l'ICTA	May 15, 2023	May 19, 2023	5 days	100	8FS
10	Refinament del conjunt seleccionat	May 22, 2023	May 24, 2023	3 days	100	9FS
11	Model de detecció d'imatges	May 25, 2023	Jun 28, 2023	25 days	100	
12	Investigació de versions	May 25, 2023	May 29, 2023	3 days	100	
13	Cerca d'implementacions de YOLO V8	May 30, 2023	May 31, 2023	2 days	100	12FS
14	Entrenament del model amb diferents datasets	Jun 01, 2023	Jun 21, 2023	15 days	100	13FS
15	Refinament del model seleccionat	Jun 22, 2023	Jun 28, 2023	5 days	100	14FS
16	Documentació	Jun 28, 2023	Jun 30, 2023	3 days	100	
17	Actualitzar diagrames	Jun 28, 2023	Jun 28, 2023	1 day	100	
18	Actualitzar requisits funcionals	Jun 28, 2023	Jun 30, 2023	3 days	100	
19	Actualitzar informe final	Jun 28, 2023	Jun 30, 2023	3 days	100	

Fig. 2: Taula amb les tasques que conformen la planificació del treball.

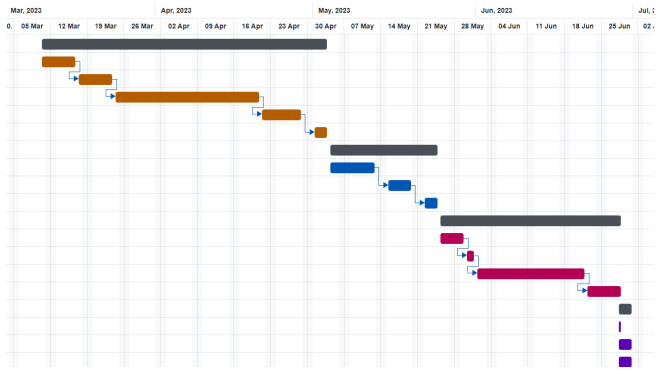


Fig. 3: Diagrama de Gantt del projecte.

4 Desenvolupament

Per tal de mantenir una certa consistència al present document, crec convenient estructurar aquesta secció de la mateixa forma que la segona secció (Estat de l'art). És a dir, la part de desenvolupament comptarà amb les següents subseccions: interfície gràfica, Dataset i model de detecció d'imatges.

Aquesta secció farà un seguiment del desenvolupament del projecte utilitzant com a referència la planificació presentada a la secció anterior i les entregues d'informes.

Per a finalitzar la introducció a la present secció, indicar que la documentació generada (document de requisits funcionals i diagrama de casos d'ús) es troba disponible per a la seva consulta a la secció "Apèndix" d'aquest informe (A3 i A4 respectivament).

4.1 Creació de la interfície gràfica

Per a començar, cal parlar del període de temps que existeix entre l'entrega de l'informe inicial i el primer informe de seguiment. Aquesta franja de temps ha estat principalment dedicada a la creació d'una interfície gràfica amb la llibreria PyQt 6, que permet a l'usuari l'etiquetatge d'imatges fent servir com a base un projecte de GitHub de l'usuari Robert Brada. [12]

La nostra GUI permet pujar fotos seleccionant la capeta contenidora i la creació de rectangles de selecció per a indicar els elements a etiquetar. Per a completar aquest etiquetatge, s'ha inclòs la funcionalitat d'exportar les dades d'aquests rectangles en format Pascal VOC o YOLO (a elecció de l'usuari) per a emprar-les futurament a l'entrenament del nostre algoritme de detecció.

A continuació, voldria indicar que l'estructura de la interfície d'etiquetatge consta de tres finestres:

- Finestra de selecció de la carpeta contenidora de les imatges. La Figura 4 mostra l'aspecte d'aquesta.
- Finestra principal d'etiquetatge. Permet visualitzar i navegar per les imatges de la carpeta seleccionada. La navegació es realitza amb les fletxes de direcció del teclat. Aquesta pantalla inclou les principals funcionalitats del nostre programa. La Figura 5 correspon a aquesta finestra i la Figura 6 mostra com es visualitzen els rectangles de selecció.
- Finestra d'exportació. Permet seleccionar la carpeta contenidora dels fitxers resultants del procés d'etiquetatge. El format d'aquests arxius serà Pascal VOC o YOLO segons l'elecció prèvia de l'usuari. L'aspecte d'aquesta pantalla és quasi idèntic al de la primera finestra d'aquesta llista.

Després d'aquesta breu introducció a l'estructura, és hora de profunditzar als aspectes més tècnics del seu funcionament. Cal dir, primerament, que els elements utilitzats per a la creació d'aquestes finestres són elements bàsics ja existents a PyQt 6 menys quan parlem dels rectangles de selecció. Aquests rectangles són objectes d'una classe personalitzada derivada d'una classe ja existent a aquesta llibreria (QRubberBand).

A continuació, parlarem del tractament que es fan de les imatges per a la seva visualització en la finestra d'etiquetatge. Totes les imatges es redimensionen a 1500 x 1125 i es calculen les ràtios de redimensió tenint en compte la mida original d'aquestes. Aquests valors són importants a l'hora d'escalar les coordenades dels rectangles de selecció a la mida original de les imatges. Per tal d'assignar una etiqueta a una selecció, només serà necessari fer doble clic a sobre del rectangle i escriure al camp de text que es mostrarà el nom de l'etiqueta.

Per a finalitzar la subsecció referent a la interfície,

voldria indicar que un dels aspectes més rellevants del seu funcionament és que tota la informació dels rectangles de selecció es manté guardada a un diccionari de Python. La clau d'aquest diccionari correspon amb el número d'imatge i els valors són objectes de la classe QRubberBand mencionada anteriorment. La importància d'aquesta elecció de disseny radica en l'etapa d'exportació, ja que permet modificar amb pocs canvis les dades a exportar perquè correspongui amb el format seleccionat per l'usuari.

La diferència en l'estructura d'aquests formats d'exportació es pot observar a les Figures 7 i 8 (que representen el format Pascal VOC i YOLO respectivament).

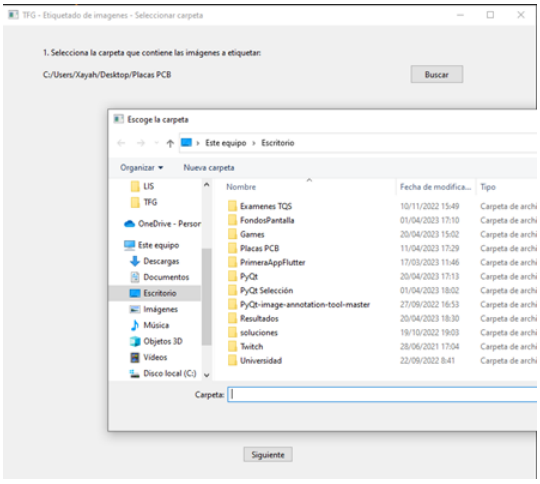


Fig. 4: Vista inicial de la finestra de selecció de carpeta.

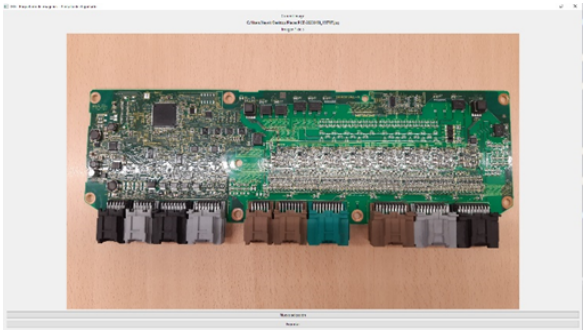


Fig. 5: Vista inicial de la finestra d'etiquetatge.

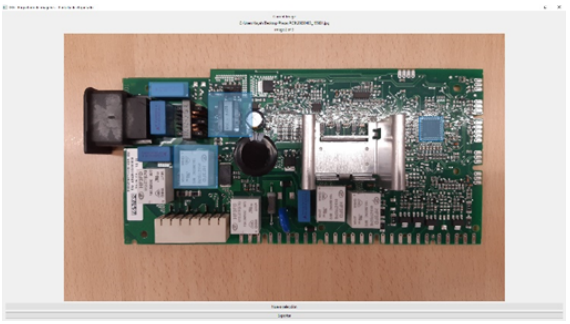


Fig. 6: Exemple de com es mostren les seleccions.

```
<?xml version="1.0"?>
- <annotation>
    <folder>Placas PCB</folder>
    <filename>20230403_155831.jpg</filename>
    <path>C:\Users\Xayah\Desktop\Placas PCB\20230403_155831.jpg</path>
  - <source>
      <database>Unknown</database>
    </source>
  - <size>
      <width>4608</width>
      <height>3456</height>
      <depth>3</depth>
    </size>
    <segmented>0</segmented>
  - <object>
      <name>Chip1</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>1505.28</xmin>
        <ymin>617.47</ymin>
        <xmax>1966.08</xmax>
        <ymax>1078.27</ymax>
      </bndbox>
    </object>
  - <object>
      <name>Chip4</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>1145.85</xmin>
        <ymin>1158.14</ymin>
        <xmax>1606.64</xmax>
        <ymax>1618.94</ymax>
      </bndbox>
    </object>
  - <object>
      <name>Chip3</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>3707.9</xmin>
        <ymin>869.37</ymin>
        <xmax>3969.02</xmax>
        <ymax>1118.2</ymax>
      </bndbox>
    </object>
  - <object>
      <name>Chip2</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>884.73</xmin>
        <ymin>503.8</ymin>
        <xmax>1299.45</xmax>
        <ymax>663.54</ymax>
      </bndbox>
    </object>
</annotation>
```

Fig. 7: Exemple de fitxer de dades en format Pascal VOC.

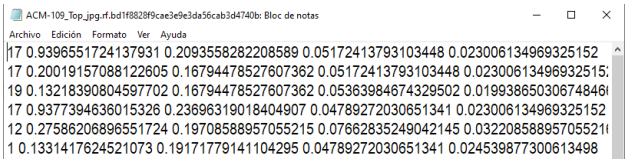


Fig. 8: Exemple de fitxer de dades en format YOLO.

4.2 CERCA I ELECCIÓ DEL DATASET

La cerca i elecció del conjunt d'imatges a emprar compta amb la seva pròpia subsecció a causa de la importància que té en el posterior entrenament del nostre model. Un dataset de qualitat és imprescindible per a poder aconseguir un model precís. Dit això, en aquesta

subsecció s'indicarà breument que fa falta per dir que un conjunt d'imatges és de qualitat (criteris que personalment he utilitzat per aprofitar o descartar opcions) i es parlarà dels principals problemes a l'hora d'obtenir datasets.

Per a començar, tractarem els factors que solen identificar un bon dataset en l'àmbit que ens ocupa. En la meva experiència desenvolupant aquest projecte crec que aquests factors són els següents:

- Un nombre d'imatges preferiblement superior a 200.
- Un etiquetatge dels elements correcte i complet. A la Figura 9 es pot apreciar una imatge corresponent a una placa PCB correctament etiquetada (imatge extreta del conjunt escollit per al projecte).
- Varietat en el tipus de plaques PCB que es mostren a les imatges.
- Preferiblement que existeixi més d'una imatge per tipus de placa i que es jugui amb quantitat de llum o angles diferents de foto.
- És important que les imatges que conformen el dataset no siguin "*frames*" de vídeo. El perquè de l'aplicació d'aquest criteri s'indica en la secció de resultats.
- Un etiquetatge de classes balancejat. El nombre d'instàncies d'una classe té un gran efecte sobre l'AP mitjà d'aquesta, és a dir, resulta més fàcil entrenar la detecció dels components que més apareixen etiquetats al conjunt d'entrenament.

Després de veure els criteris utilitzats a l'hora d'escollir datasets, és crucial parlar de les dificultats existents a l'hora de trobar bons conjunts d'imatges. Un dels principals problemes és la proliferació de datasets a pàgines com Roboflow que són creats per projectes personals o per aprendre. Existeixen una quantitat enorme d'aquest tipus de conjunts a causa de la popularitat creixent de la intel·ligència artificial, però solen ser conjunts de baixa qualitat que fan més complexa la cerca. Un altre factor rellevant és la mancança de conjunts públics de qualitat en l'àmbit de detecció de components electrònics. Aquest últim punt és causat pel grau d'especificació de la tasca actual juntament amb la complexitat que representa la detecció d'elements de mida reduïda (parlem, a vegades, de mides inferiors a tres píxels en la imatge).

Una vegada presentades les dificultats, és moment d'indicar quin és el conjunt escollit. La franja de temps entre el primer informe de seguiment i el tercer ha sigut, principalment, per provar diferents conjunts entrenant un model de YOLO V8. Els resultats de les proves amb alguns datasets es troben a la secció de resultats. El conjunt escollit és part d'una iniciativa patrocinada per Intel anomenada RF100 [13] i és producte de la col·laboració d'un grup d'usuaris de Roboflow. Aquest conjunt anomenat "*printed circuit board Computer Vision Project*" [14] conté un total de 672 imatges i és capaç de detectar un total de 23 classes diferents.

Per a finalitzar aquesta subsecció és important destacar que el conjunt d'imatges escollit ha sigut ampliat utilitzant les imatges proporcionades per l'ICTA. L'etiquetatge d'aquestes noves imatges s'ha realitzat amb l'eina implementada per a aquest treball.

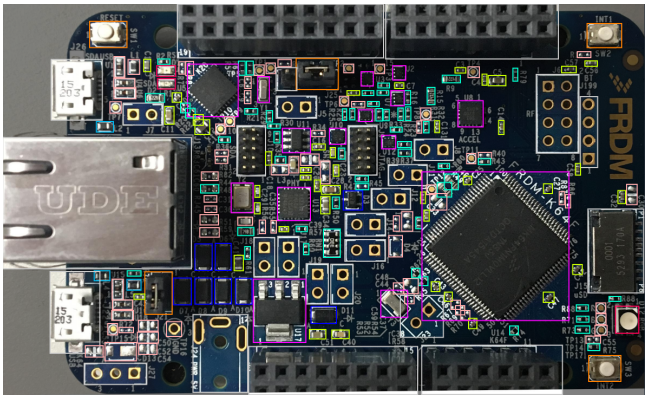


Fig. 9: Exemple d'imatge amb un etiquetatge complet i correcte.

4.3 Entrenament del model de detecció

Per a començar aquesta subsecció, parlarem d'altres versions de YOLO. El motiu d'aquesta decisió és que, inicialment, es feien servir les versions 5 i 7 d'aquest model per dur a terme els entrenaments. Els factors claus darrere d'aquest canvi de versió en l'últim tram del projecte són varis i ho considero un punt crucial a l'hora de parlar del desenvolupament.

Després d'aquesta introducció, és necessari definir quins són els factors que més rellevància tenen a l'hora d'entrenar un model de detecció d'imatges. Els factors en qüestió són els següents:

- Resolució de la imatge. Grans resolucions augmenten el consum de memòria de la GPU. Aquest increment de consum es pot compensar, fins a cert punt, reduint la mida del "*batch*".
- Mida del "*batch*". Quantitat d'imatges que es tracten abans d'actualitzar el model. Un conjunt amb imatges de baixa resolució permet tenir una major mida d'aquest paràmetre. Té un efecte directe en el temps d'entrenament.
- Nombre de classes del dataset. Utilitzar un conjunt complex amb moltes classes i instàncies requereix major memòria.
- Memòria de la GPU del equip emprat per l'entrenament. Aquest és el factor més determinant. Una gràfica potent permet reduir el temps d'entrenament o augmentar el nombre d'èpoques (té una repercussió directa a la precisió final que tindrà el model).
- Èpoques. Nombre de cops que l'algoritme processarà el conjunt d'entrenament. Té un efecte directe sobre el temps del procés i sobre el grau de refinament del model resultant.

Una vegada introduïts aquests factors, entrem als aspectes més tècnics d'aquesta part del treball. A les fases inicials d'aquest projecte feia servir el meu propi equip per a portar a cap els entrenaments. El processament d'imatges requereix la utilització de la GPU i, en el meu cas, aquesta és una NVIDIA 1050 TI de 4 GB. Aquesta baixa capacitat de memòria a la gràfica resultava en la impossibilitat d'entrenar el model fent ús de datasets amb bona resolució o d'aplicar un nombre mitjanament gran d'èpoques. Els resultats en aquesta fase no eren

satisfactoris i el procés d'entrenament era lent.

Per una altra banda, en la franja de temps entre el segon i el tercer informe de seguiment, es comença a fer ús de Google Colab. Aquesta eina de Google ens proporciona un entorn d'execució remot que permet la utilització d'una gràfica NVIDIA T4 de 16 GB sense subscripció. L'augment en els recursos disponibles va portar aquesta experimentació amb YOLO a un nou nivell, però també tenia inconvenients. Els comptes gratuïts de Colab tenen un límit de temps d'execució i, quan aquest temps acaba, avorta els processos actius i esborra els resultats. Els resultats obtinguts en aquesta fase del projecte estaven lluny de ser considerats bons i sorgeix la necessitat de fer canvis en la forma de treballar.

Després de veure les limitacions trobades, parlarem del període de temps entre el darrer informe de seguiment i l'entrega final. En aquest darrer tram del projecte, es pren la decisió de fer el canvi a YOLO 8 a causa de la millora de rendiment que presenta respecte a anteriors versions. Aquesta millora permet aprofitar d'una forma més productiva el temps d'execució a Google Colab. Un altre canvi en la forma de treballar és l'estratègia d'entrenament. Inicialment, executava procesos amb un nombre d'èpoques gran i, actualment, es realitzen unes 100 iteracions per dia amb el fi d'assolir un refinament continu del model. Aquest refinament consta d'entrenaments amb poques èpoques, però cada vegada amb més resolució. Una resolució gran és el que permet una millor identificació i detecció d'aquells components de mida més reduïda.

Per a finalitzar aquesta secció, és rellevant esmentar que la implementació de YOLO utilitzada és la proporcionada per Ultralytics. [15] Aquesta empresa té com a objectiu fer més accessible la intel·ligència artificial i l'optimització de models per a obtenir algoritmes més eficients.

5 Resultats

Aquest últim apartat del document estarà principalment centrat en els resultats assolits dels entrenaments i deteccions del model YOLO. El motiu d'aquesta decisió és que la part pertinent a la interfície gràfica es mostra en les Figures de la secció 4.1 i, a més, la detecció de components resulta més critica a l'hora d'assolir el nostre objectiu.

Una vegada especificat això, és necessari introduir una sèrie de mètriques que es fan servir per mesurar, entre altres, com és de bo el model actual. Aquests factors a tenir en compte són els següents:

- Grau de confiança. És un número entre 0 i 1 que indica en quin percentatge està segur el model del fet que la seva predicció sigui correcta. Aquest paràmetre s'utilitza principalment per filtrar les deteccions, generalment establint un cert grau de confiança (sol ser d'un 25% per defecte).
- IoU (*Intersection over Union*). Mètrica emprada per a avaluar la precisió de les deteccions. Es calcula comparant la intersecció del rectangle predit i el quadre delimitador que representa la ubicació real d'un objecte. El rang de valors del IoU és entre 0 i 1, sent xifres properes a 1 indicador d'una detecció precisa (per defecte, el llindar establert per YOLO és d'un 65%).
- mAP (*mean Average Precision*). Mètrica usada en varietat de models de detecció que serveix per avaluar el rendiment general en la detecció de múltiples classes. Aquest valor combina les mètriques de precisió / "*recall*" i es calcula fent la mitjana de les mitjanes de totes les classes. Xifres properes a 1 indican un major rendiment i valors propers a 0 són indicatius de dificultat a l'hora de detectar certs elements.

Després de conèixer aquests conceptes, és hora d'avaluar els resultats obtinguts. Primerament, parlarem de les diferències observades a l'hora de treballar amb diferents conjunts d'imatges. Cal indicar que els valors que es presenten a la taula de la Figura 10 han sigut aconseguits emprant YOLO V7.

Dataset	Imatges	Classes	*Epochs*	mAP
1	690	16	100	10%
2	1113	9	100	20%
3	270	22	100	24%
4	672	23	100	41%

Fig. 10: Taula que mostra el mAP assolit en diferents execucions emprant diferents datasets.

Al llarg del desenvolupament d'aquest projecte han sigut testejats més de set conjunts d'imatges diferents. El motiu pel qual només es mostren els resultats d'aquests 4 és per explicar els problemes que poden presentar els datasets i que afecten de gran manera al mAP resultant. Els factors que poden afectar el rendiment del model es llisten a continuació:

- Cas del conjunt 1. Les imatges que conformen aquest dataset corresponen a components ja extrets de les plaques. Un altre factor que justifica un resultat de només el 10% és que una gran part de les imatges són d'una baixa qualitat.
- Cas del conjunt 2. Un exemple perfecte de què un número més gran d'imatges no és necessàriament sinònim d'un major mAP. La majoria de les fotos són "*frames*" d'un mateix vídeo i, per tant, corresponen a la mateixa placa. L'altre factor que fa d'aquest conjunt un dataset inadequat és que no hi ha diferència entre les imatges corresponents a les parts d'entrenament i validació. Aquests fets resulten a dir que aquest mAP obtingut del 20%, encara sent un percentatge baix, sigui un resultat edulcorat.
- Cas del conjunt 3. Aquest dataset compta amb una bona varietat de plaques, però són poques imatges i l'etiquetatge no és 100% complet.
- Cas del conjunt 4. Un nombre d'imatges considerable amb un etiquetatge correcte i molt complet. Aquest és el dataset seleccionat per al nostre projecte.

Un cop observats aquests resultats i coneixent els principals indicatius d'un conjunt de baixa qualitat, és moment de veure el procés de refinament del dataset seleccionat. Abans de veure les imatges corresponents és necessari explicar com ha sigut realitzat aquest procés dia a dia. Tenint en compte els recursos computacionals disponibles, han sigut portades a cap múltiples execucions amb un nombre petit d'èpoques cada dia. Aquesta forma de refinament permet una millora contínua del model sense arribar al límit de temps de còmput que imposa Google Colab. Un altre factor a destacar és l'augment progressiu de la resolució de les imatges en cada entrenament. Aquest modus operandi te com a objectiu millorar la precisió a l'hora de detectar aquells components de mida reduïda com, per exemple, les resistències. Les Figures 11, 12 i 13 mostren els resultats d'aquests entrenaments continus sobre el mateix model.

Model summary (fused): 168 layers, 11134485 parameters, 0 gradients, 28.5 GFLOPs

Class	Images	Instances	Box(P	R	mAP50	mAP50-95)
all	80	18602	0.494	0.349	0.385	0.26
Button	80	60	0.81	0.667	0.754	0.503
Capacitor Jumper	80	3631	0.238	0.206	0.148	0.0656
Capacitor	80	3630	0.357	0.0979	0.175	0.0905
Clock	80	10	0.701	1	0.938	0.712
Connector	80	714	0.538	0.619	0.576	0.411
Diode	80	26	0.407	0.154	0.207	0.177
EM	80	38	0.852	0.368	0.542	0.386
Electrolytic Capacitor	80	166	0.772	0.843	0.862	0.679
Ferrite Bead	80	2	0	0	0	0
IC	80	1026	0.59	0.75	0.689	0.446
Inductor	80	42	0.158	0.0952	0.152	0.122
Jumper	80	46	0.505	0.478	0.512	0.204
Led	80	120	0.812	0.505	0.64	0.358
Pads	80	100	0.44	0.28	0.319	0.183
Pins	80	200	0.798	0.49	0.571	0.44
Resistor Jumper	80	3341	0.215	0.124	0.112	0.0468
Resistor Network	80	3927	0.325	0.0385	0.109	0.0591
Resistor	80	597	0.448	0.0151	0.166	0.0954
Switch	80	44	0.763	0.818	0.862	0.744
Test Point	80	52	0.253	0.0962	0.145	0.0675
Transistor	80	748	0.379	0.392	0.337	0.16
Unknown Unlabeled	80	76	0	0	0.0505	0.0249
iC	80	6	1	0	0	0

Fig. 11: Taula que mostra el mAP obtingut amb 50 èpoques i resolució 928x928.

Model summary (fused): 168 layers, 11134485 parameters, 0 gradients, 28.5 GFLOPs

Class	Images	Instances	Box(P	R	mAP50	mAP50-95):
all	80	18602	0.473	0.421	0.438	0.299
Button	80	60	0.789	0.733	0.79	0.651
Capacitor Jumper	80	3631	0.28	0.305	0.192	0.087
Capacitor	80	3630	0.419	0.0996	0.198	0.111
Clock	80	10	0.443	1	0.82	0.621
Connector	80	714	0.564	0.606	0.588	0.43
Diode	80	26	0.347	0.308	0.337	0.244
EM	80	38	0.838	0.632	0.777	0.544
Electrolytic Capacitor	80	166	0.743	0.916	0.914	0.685
Ferrite Bead	80	2	0	0	0	0
IC	80	1026	0.621	0.645	0.638	0.425
Inductor	80	42	0.283	0.113	0.167	0.151
Jumper	80	46	0.851	0.609	0.743	0.265
Led	80	120	0.649	0.558	0.622	0.432
Pads	80	100	0.493	0.51	0.5	0.298
Pins	80	200	0.823	0.68	0.723	0.553
Resistor Jumper	80	3341	0.219	0.284	0.16	0.0648
Resistor Network	80	3927	0.361	0.0509	0.127	0.0692
Resistor	80	597	0.453	0.121	0.228	0.12
Switch	80	44	0.875	0.864	0.904	0.779
Test Point	80	52	0.46	0.308	0.328	0.181
Transistor	80	748	0.364	0.334	0.275	0.141
Unknown Unlabeled	80	76	0	0	0.0456	0.0178
iC	80	6	0	0	0	0

Fig. 12: Taula que mostra el mAP aconseguit amb 50 èpoques més i resolució 1024x1024.

Model summary (fused): 168 layers, 11134485 parameters, 0 gradients, 28.5 GFLOPs

Class	Images	Instances	Box(P	R	mAP50	mAP50-95):
all	80	18602	0.518	0.424	0.45	0.316
Button	80	60	0.78	0.6	0.709	0.595
Capacitor Jumper	80	3631	0.272	0.288	0.194	0.0912
Capacitor	80	3630	0.409	0.128	0.207	0.12
Clock	80	10	0.542	1	0.849	0.699
Connector	80	714	0.57	0.634	0.597	0.43
Diode	80	26	0.382	0.308	0.374	0.298
EM	80	38	0.85	0.596	0.671	0.504
Electrolytic Capacitor	80	166	0.881	0.88	0.924	0.75
Ferrite Bead	80	2	0	0	0	0
IC	80	1026	0.608	0.685	0.654	0.442
Inductor	80	42	0.406	0.0952	0.123	0.114
Jumper	80	46	0.831	0.609	0.775	0.341
Led	80	120	0.853	0.642	0.762	0.518
Pads	80	100	0.457	0.46	0.445	0.253
Pins	80	200	0.821	0.72	0.754	0.575
Resistor Jumper	80	3341	0.227	0.346	0.185	0.081
Resistor Network	80	3927	0.352	0.0703	0.151	0.0833
Resistor	80	597	0.526	0.0763	0.229	0.123
Switch	80	44	0.791	0.909	0.927	0.821
Test Point	80	52	0.518	0.346	0.376	0.202
Transistor	80	748	0.376	0.29	0.29	0.152
Unknown Unlabeled	80	76	0.452	0.0652	0.145	0.0705
iC	80	6	0	0	0	0

Fig. 13: Taula que mostra el mAP assolit amb 25 èpoques addicionals i resolució 1280x1280.

6 Conclusions

Com es pot observar en les Figures anteriors, el rendiment general del model progressa adequadament. Les classes que presenten una precisió més baixa són aquells elements de mida reduïda, però els principals components de les plaques PCB compten amb precisions que arriben fins i tot al 96%. Les diferents proves realitzades i els resultats mostrats a la secció anterior demostren que amb més capacitat de còmput i temps, es pot assolir un model de detecció en temps real i amb una precisió considerable.

La conclusió després de finalitzar aquest projecte és que els models actuals de detecció d'imatges són tecnologies potents que, gràcies a comunitats com Ultralytics o Roboflow, poden continuar creixent, millorant i esdevenint cada cop més accessible.

Agraïments

M'agradaria agrair, primerament, a Delfina Esquerra Rejo (la meva mare). Gràcies al seu esforç he pogut arribar a on soc i presentar aquest treball com a culminació dels meus estudis. Gràcies per ser també un exemple de superació.

M'agradaria també agrair a la meva parella, Sofia Juan Chiner, pel seu suport als pitjors moments tant de la vida com en l'elaboració del present treball. Aprenc cada dia molt de tu com a persona i em fas millor.

Finalment, vull agrair als amics de sempre. Gràcies a Noe, Rodri e Ygor per estar presents sempre que ha sigut necessari al llarg de la carrera i de la vida. Heu fet d'una experiència que, a vegades, ha sigut massa dura fos molt millor. Sou també una llum a la foscor.

Bibliografia

[1] América RETAIL: Automatización de procesos con Inteligencia Artificial. https://www.america-retail.com/transformacion-digital/automatizacion-de-procesos-con-inteligencia-artificial-la-revolucion-tecnologica-y-humana-de-las-empresas/ , 2022.

[2] *Latest Global E-Waste Statistics And What They Tell Us.* https://theroundup.org/global-e-waste-statistics/, 2023.

[3] Agbogbloshie. https://es.wikipedia.org/wiki/Agbogbloshie, 2023.

[4] K. Weinert, V. Pandey, S. Naranjo Corona, A. Danielewski. *Autonomous waste characterization in environmentally conscious decision making*.
[5] Extreme Programming For One. https://wiki.c2.com/?ExtremeProgrammingForOne, 2014.
[6] Roboflow. https://www.roboflow.com, 2023.
[7] G. Mahalingam, K. M. Gay and K. Ricanek. PCB-METAL: *A PCB Image Dataset for Advanced Computer Vision Machine Learning*. https://ieeexplore.ieee.org/abstract/document/8757928, 2019.
[8] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg. *SSD: Single Shot MultiBox Detector*. https://www.arxiv.org/abs/1512.02325, 2016.
[9] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. https://www.arxiv.org/abs/1506.01497, 2016.
[10] Joseph Edmonton, Santosh Dybala, Ross Girshick, Ali Farhadi. *You Only Look Once: Unified, Real-Time Object Detection*. https://www.arxiv.org/abs/1506.02640, 2016.
[11] Yingqi Lu, Bo Yang, Yichun Gao, Zhenming Xu. *An automatic sorting system for electronic components*. https://www.sciencedirect.com/science/article/pii/S0956053X21005560, 2022.
[12] *PyQt-image-annotation-tool*, Robert Brada (robertbrada on GitHub). https://github.com/robertbrada/PyQt-image-annotation-tool , 2022.
[13] Roboflow 100: A New Object Detection Benchmark. www.rf100.org, 2022.
[14] Roboflow 100, *printed circuit board Dataset*. https://universe.roboflow.com/roboflow-100/printed-circuit-board, 2023.
[15] Ultralytics YOLOv8: *The State-of-the-Art YOLO Model*. https://www.ultralytics.com/yolov8, 2023.

APÈNDIX

A1. Taula de tasques del projecte amb dates i temps de duració (en dies)

ListaDeTareas.PNG

A2. Diagrama de Gantt del projecte

DiagramaDeGantt.PNG

A3. Enllaç al document d'objectius i requisits funcionals del projecte

TFG-Documento técnico.docx

A4. Enllaç al diagrama de casos d'ús corresponent als requisits funcionals del projecte

CasosDeUso.png