FlutterCV: Flutter App development with Computer Vision modules

Juan Carlos Soriano Valle

Resum—Resum del projecte, màxim 10 línies. ....
....
....
....
....
....
....
....
....
....

Paraules clau—Paraules clau del projecte, màxim 2 línies. ....
....

Abstract—Versió en anglès del resum. ....
....
....
....
....
....
....
....
....
....

Index Terms—Versió en anglès de les paraules clau. ....
....

1. Introducció - Context del treball

En els darrers anys, el sector de l’oci té un component que creix de forma exponencial: el videojoc. Aquest fet té moltes causes: van ser un salvavides per la gent durant la pandèmia, cada vegada apareixen més en tot tipus d’anuncis, està creixent la capacitat de fer videojocs més reals o més òptims per poder jugar en dispositius de baix-mig nivell, etc.

Un dels grans videojocs que van ser claus durant la pandèmia va ser l’*Animal Crossing: New Horizons* [1]. Aquest és un videojoc que va sortir just va començar la quarantena global, el 20 de Març del 2020, i gràcies a aquesta situació juntament amb l’estil de videojoc i el temps que portaven els jugadors esperant una nova entrega d’aquesta saga, va acabant sent un exit en vendes [2]. Tant és així que va aconseguir vendre un total de 11,7 milions d’unitats durant els primers 10 dies al mercat a tot el mon. A data de febrer de 2022, es situa en les 37,62 milions d’unitats, havent venut 32,63 milions només en el primer any (fins març de 2021). A més de totes les vendes, va ser nominat a tots els premis de millor joc de l’any 2020 i 2021 [3], guanyant premis com “Japan Game Awards – Game of the Year”, “The Game Awards 2020 – Best Family Game”, entre d’altres.

Com que aquest és un videojoc que tracta de tenir el teu personatge que va fent la seva vida dia a dia amb diferents tasques, objectius i col·leccionables, arriba un moment en el progrés del joc que el jugador no pot tenir memoritzat l’estat de totes les tasques i catàleg del seu inventari. És per aquest fet que ja fa un temps s’han posat de moda unes aplicacions o programes que t’ajuden a complir aquest objectiu afegint al jugador eines concretes per a cada videojoc. Aquestes eines reben el nom de “Companion”. En el cas de l’*Animal Crossing*, una app *Companion* ajudaria l’usuari a mantenir un estat actualit-

- *E-mail de contacte: juancarlos.soriano@autonoma.cat*
- *Menció realitzada: Enginyeria de Computació*
- *Treball tutoritzat per: Felipe Lumbreras Ruiz (departament de Ciències de la Computació)*
- *Curs 2021/22*

zat de la seva col·lecció d'animals i obres d'art que pot recopilar al museu de la seva ciutat, tenir informació detallada de cada animal i quan es poden obtenir (meteorològicament, estació de l'any, hora del dia, etc.).
Aquestes tasques ja les fan aplicacions que s'han publicat, però des d'una perspectiva d'un jugador "veterà", trobo que hi ha moltes tasques que no s'han fet encara i que millorarien molt l'experiencia d'usuari del videojoc. Entre elles, ajudaria a l'usuari a reconèixer si l'obra d'art que un personatge determinat del videojoc intenta vendre és verdadera o és una imitació quasi perfecta. Aquest personatge té un gran nombre d'obres d'art per vendre, però només apareix 1 vegada una o dues setmanes, i de les 5 peces que et ven, normalment només 1 és verdadera.
Aquest projecte doncs implementarà funcions que no s'han implementat fins ara en les apps Companion d'aquest joc amb l'ajuda de mòduls de visió per computació dintre de l'app mòbil.

2. OBJECTIUS DEL PROJECTE

Tal com s'ha comentat breument en l'anterior punt, aquest projecte té com a objectiu aprofitar funcions de la Visió per Computació per ajudar a l'experiència de joc de l'"Animal Crossing". El desenvolupament tindrà una forma seqüencial, a on a cada iteració s'implementarà una funcionalitat diferent, i no es començarà la implementació de les següents funcions fins que l'actual no arribi a un estat de *Minimum viable product*. Aquests subobjectius són:

2.1 Reconeixement de l'obra d'art

El primer objectiu que tindrà aquest projecte serà el reconeixement de l'obra d'art que té el jugador en la pantalla de la seva consola.

Fig. 1. Vista prèvia abans de comprar l'obra "Dama amb un ermini"

En aquest cas el videojoc és per a la consola Nintendo Switch [4], que té una pantalla de 6,2" amb tecnologia LCD. L'usuari haurà de fer una fotografia de la seva pantalla amb el telèfon mòbil, que a través de l'aplicació li mostrarà quina obra és i si està falsificada o no. Aquesta casuística fa que les imatges que es processaran no siguin iguals, canviant aspectes com la il·luminació, la perspectiva i la posició de la pantalla de la consola en la fotografia. A la figura 1 es mostra un exemple de com es veuria dintre de la sessió del joc.

2.2 Reconeixement de l'insecte o peix (OCR)

En aquesta funcionalitat, el programa tractarà de reconèixer quina espècie acaba d'atrapar l'usuari, en una pantalla similar a la que es mostra en la figura 2. Les espècies poden ser peixos, criatures marines i insectes. Quan s'hagi esbrinat de quina espècie es tracta, el programa et dirà si l'usuari ja té aquest animal registrat al museu. Aquesta funcionalitat es basa en processar el text que diu el personatge quan atrapa a qualsevol animal.

Fig. 2. Frase de pesca d'una tonyina

2.3 Reconeixement de l'insecte o peix (Object Recognition)

En aquest punt es treballarà sobre el mateix problema que el punt anterior, amb la diferència que en comptes d'interpretar el missatge de captura, es basarà en la pantalla de l'enciclopèdia que té el jugador. Aquesta funcionalitat estarà pensada en la situació en què un jugador té una espècie en concret que o no surt en l'estació actual entre molts exemples. També és idoni per un jugador que està en un punt molt avançat del progrés del joc, i prefereix registrar tots els espècimens que té directament des de la seva col·lecció.

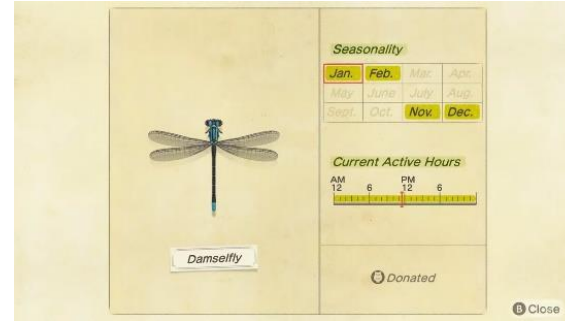


Fig. 3. Mostra de l'entrada de l'enciclopedia d'un insecte

A la pantalla de cada animal figura una imatge la qual es posiciona de manera diferent per a cada animal, apareixent informació addicional tal com els mesos on apareix, a quines hores apareix, tal com es mostra a la figura 3.

2.4 Mòduls VC en un entorn de producció

Tots els mòduls que es volen desenvolupar ho faran des d'un ordinador, però l'objectiu és incorporar-los a un entorn gràfic i interactiu com és una aplicació mòbil. Aquest fet té les dificultats del fet que un telèfon mòbil no té la potència d'un ordinador. A més, quan es desplega algun programa d'aquest estil, no sol tenir cap interfície gràfica, sinó que s'executen directament des de consola o amb alguna comanda especial.

Per tant, un propòsit d'aquest projecte és implementar aquests mòduls en un entorn amigable per a un usuari final en un entorn de producció, fora de l'entorn de desenvolupament.

2.5 Desenvolupament en entorns diferents

Tal com s'ha exposat a l'anterior punt, el dispositiu final on es visualitzarà tot el programa i que servirà com a punt d'introducció de l'input és un dispositiu mòbil, a on s'executarà l'aplicació que es vol desenvolupar. Per això, s'ha de tenir en compte la potència de còmput entre un ordinador i un telèfon mòbil. En termes dels mòduls de visió per computació que es volen implementar en aquest projecte, això es tradueix en el fet que es necessita reduir al màxim l'impacte computacional del programa.
L'aplicació s'executarà en Flutter, component que es profunditzarà més endavant, però la part de backend de l'aplicació, on estarà tota la massa computacional, tindrà 2 variants. El desenvolupament es dividirà en 2 "sistemes":

- **Model entrenat prèviament a l'ordinador i importat a l'aplicació mòbil.** Aquesta variant es basa a entrenar un model pel mòdul de deep learning per la part d'image recognition, per després importar aquest model al codi de l'aplicació perquè només s'encarregui de fer les prediccions d'una forma òptima.
- **Part backend a un entorn Cloud.** Aquesta variant utilitzarà la computació al núvol tal com Google Cloud o AWS per l'execució de tots els mòduls de Visió per Computació. D'aquesta manera l'aplicació només s'encarregarà de fer trucades a funcions i rebre la seva resposta.

Per tant, es durà a terme un desenvolupament en paral·lel de dues aproximacions per a un mateix problema, com és el d'incorporar funcions de visió per computació a un entorn tan poc tractat en la menció de computació com és la d'una aplicació final, ja sigui una app mòbil, un programa gràfic d'escriptori, etc. Però en un entorn de producció, lluny de consola de comandes il·legibles pels usuaris finals.

2.6 Preparació dels datasets

Per últim però no menys important, els models a entrenar ho han de fer sobre un conjunt de dades. En aquest projecte el tema és tant concret i tancat que no hi ha cap data set categoritzat i etiquetat. Per tant, una de les primeres tasques a l'hora de fer la part de desenvolupament del codi i els models serà la creació d'un dataset per a cada model que s'hagi de crear.
Principalment hi haurà un per les obres d'art i un altre per els diferents animals que hi ha en el joc.

3. Estat de l'Art

Els diferents mòduls de Visió per Computació que conformen el projecte són de temàtiques diferents, d'aquesta manera, en aquesta secció es tractaran diferents temes en comptes de centrar-se en un de sol.

3.1 Object Detection

Aquesta tasca es basa en detectar instàncies d'objectes d'una classe determinada en una imatge o un vídeo. Dintre dels mètodes més coneguts i utilitzats es poden trobar 2 tipus:

- **Mètodes "One-Stage":** Aquests mètodes es basen en prioritzar la velocitat de la inferència per tal de tenir resultats més ràpids, sacrificant la precisió del mètode. Entre els mètodes que entren dintre d'aquest rang es troben el YOLO [5], SSD [6] i RetinaNet [7].
- **Mètodes "Two-Stage":** Aquests mètodes es basen en donar més passos ja sigui de transformacions a les dades o passades extres dels models, el que fa que augmentin la precisió de la detecció per tenir uns millors resultats finals. Exemples de mètodes d'aquest tipus són: Faster R-CNN [8], Mask R-CNN [9] i Cascade R-CNN [10].

A part d'aquests mètodes més coneguts, n'hi ha un que recentment s'està consolidant com el mètode que millors resultats aconsegueix en aquest camp. Aquest és el SwinV2-G [11], que es basa en escalar la capacitat i la resolució màxima possible amb la que es pot entrenar el model Swin Transformer. Amb aquesta nova versió s'ha aconseguit encapçalar les puntuacions dels datasets més populars en aquest camp (ADE20K [12], COCO minival i COCO test-dev [13], ImageNet [14], entre d'altres).

3.2 OCR

En aquest camp hi ha diferents aproximacions que intenten interpretar el text que conté la imatge a processar. Hi ha dues fases per tal de llegir un text en una imatge. El primer pas és detectar el text en la imatge. Entre aquestes tècniques es troben la "sliding window" que es basa en anar desplaçant una finestra per tota la imatge. Un exemple de mètode que utilitza aquesta tècnica és una CNN [15]. D'altres tècniques que existeixen són les "single shot" com la YOLO [16] a la qual es passa només una vegada la imatge i detecta el text en una regió determinada i per últim l'"EAST" [17], un mètode molt avançat i precís que fins i tot pot detectar text en un vídeo a 13fps en qualitat 720p.

L'altre pas de l'OCR és reconèixer el text que hi ha a la regió. Per això es fan servir Xarxes Neuronals tals com la CRNN [18], una combinació de xarxa convunacional i xarxa recurrent.

3.3 Aplicacions Mòbils

Aquest és un sector que evoluciona molt ràpid, on surten noves tecnologies cada poc temps i les que existeixen s'actualitzen de forma molt freqüent si no volen estancarse i desaparèixer. Entre aquestes plataformes [19] s'ha de tenir en compte el factor més important: el SO. Les que més s'utilitzen actualment en el món d'Android són:

- **JavaScript**: Aquest llenguatge de programació va ser un dels més utilitzats gràcies al gran nombre de frameworks que l'utilitzen com poden ser Angular, React, etc. Destaca la seva facilitat per crear una estructura de full-stack molt còmodament. En el cas de React, també ens permetrà programar apps per iOS.
- **Java**: D'igual manera que JavaScript, aquest és

un llenguatge molt polivalent que també permet fer una estructura full-stack.
- **Kotlin**: Al tenir suport natiu d'Android, Kotlin és la millor opció si es vol desenvolupar una aplicació en Android i es volen aprofitar totes les característiques tant del sistema operatiu com del telèfon mòbil.

En canvi, si es vol desenvolupar una app en iOS, les millors opcions són:
- **Swift**: És el llenguatge oficial d'iOS. Per aquest fet, té suport directe d'Apple, una gran avantatge al ser un sistema operatiu tan tancat. Al ser natiu, utilitza totes les funcions possibles del telèfon mòbil.

Fora de l'àmbit natiu, el framework que està creixent d'una forma exponencial és Flutter [20]. Aquest framework que ha creat Google fa servir el llenguatge Dart. La principal avantatge d'aquest sistema és que d'un mateix codi es pot crear una aplicació d'Android, d'iOS, d'escriptori de Windows i fins i tot una pàgina web.

4. METODOLOGIA

En aquest projecte s'utilitzaran diferents mètodes i eines per tal de portar una correcta gestió i desenvolupament de tot el conjunt de fases del projecte.
Pel que fa a metodologies, es farà servir una metodologia àgil tal com és SCRUM [21], amb una configuració d'sprints. La seva durada estarà determinada per les diferents entregues que hi hagi en el transcurs del TFG, amb l'excepció d'alguna funcionalitat gran que tingui moltes subtasques o períodes molt grans entre lliuraments. Això fa que els sprints tinguin una durada d'entre 2-3 setmanes, una durada òptima per tal de treballar amb aquesta metodologia.
Pel que fa a les eines que s'utilitzaran, aquestes seran: Jira [22] per la gestió i els seguiments dels sprints, Notion [23] per la documentació interactiva del progrés i experiments que es duran a terme en el transcurs del projecte, Github [24] per la creació del repositori del projecte i Toggl [25], una eina de time tracking. Es trobarà informació més detallada sobre les eines i la seva utilització en el dossier (pàgina de Notion) del projecte.

La majoria d'aquestes eines i metodologies estàn pensades per aplicar-les en projectes col·laboratius amb un equip de persones. En aquest cas, el projecte només es desenvoluparà per una persona, però és una molt bona oportunitat per tenir un primer contacte en un àmbit de projecte real.

5. PLANIFICACIÓ INICIAL DEL PROJECTE

Les diferents tasques del projecte es basen en la redacció de l'informe del treball, la documentació dels diferents temes que es tractaran en el seu transcurs i les tasques de desenvolupament de les diferents funcionalitats, tant de l'aplicació com dels mòduls de Visió per Computació.

Per tant, s'haurà de tenir en compte que en algunes etapes del projecte es faran algunes d'aquestes tasques de manera simultània.

D'una manera molt preliminar i sense tenir cap referència del temps de les tasques la planificació inicial seria aquesta:

Data	Output desitjat
06-03-22	Primer Informe i documentació tècniques de VC inicials
20-03-22	Primera versió mòdul image recognition
27-03-22	Millores mòdul Img Rec i primera versió de l'App
10-04-22	Informe progrés I i millores mòdul Img Rec
24-04-22	Img Rec acabat, primera versió OCR i millora de l'aplicació
08-05-22	OCR acabat
22-05-22	Informe progrés II i primera versió Img Rec animals
05-06-22	Desenvolupament app i Img Rec animals acabat
12-06-22	Proposta Informe Final
26-06-22	Proposta Presentació
27-06-22	Lliurament Dossier

Taula 1. Planificació inicial del projecte

Per aprofundir més en aquesta planificació, consultar a l'Apèndix 1 el Diagrama de Gantt corresponent.
Cal aclarir que aquesta planificació és aproximada i pot canviar segons el transcurs del projecte i del seu progrés.
També existeix el fet que la part *backend* de l'aplicació es desenvoluparà tant d'una forma "local" amb un model prèviament entrenat i importat al telèfon mòbil i una altra versió que tota la potència de còmput es fa a un servei Cloud.

6. PROGRÉS DEL PROJECTE I

Durant aquest període s'ha pogut seguir la planificació inicial prevista. En els següents subpunts s'explicaran en detall les tasques realitzades. A més, s'han utilitzat les eines Roboflow [26] i WandB [27] per la generació del dataset i el seguiment dels experiments. Al dossier del projecte es troba una descripció detallada de cada eina i el seu paper en el desenvolupament dels models.
Per una millor perspectiva de l'estat actual del projecte, es pot consultar a l'Apèndix 2 el diagrama de Gantt de les tasques que s'han dut a terme en aquest període.

6.1 Creació del dataset per Object Recognition

En aquest punt s'ha creat un dataset personalitzat amb imatges tretes directament del videojoc. En total, s'han aconseguit les 70 obres entre quadres i estàtues que existeixen i s'han fet múltiples fotografies per a cadascuna, canviant aspectes com la direcció de la foto, la proximitat de l'obra i la càmera, etc. La majoria de les imatges s'han capturat amb un telèfon mòbil en format horitzontal amb unes dimensions de 4000x3000 píxels. Un 2% de les imatges totals són captures de pantalla d'extractes de vídeos d'internet o una comparació directa entre 2 obres, l'original i la falsa.

Un cop aconseguit el dataset, s'han etiquetat amb boxes i classes a través de l'eina Roboflow. Un dels avantat-

ges que té aquesta eina és la possibilitat de fer un data augmentation sobre el dataset. Fent servir aquesta funció, s'ha afegit variació de brillantor, exposició i desenfocament.

El dataset consta d'unes 70 classes diferents i un mínim de 6 imatges per classe. En total, hi ha unes 446 imatges on hi ha mínim una classe representada i aproximadament 10 fotos on no hi ha cap classe. El total del dataset puja fins a les 982 imatges tenint en compte el data augmentation que s'aplica sobre el set de training.

6.2 Desenvolupament dels models d'Object Recognition

Un cop creat un model prou bo i precís per entrenar un model d'Object Recognition, s'estableix YOLOv6 com l'arquitectura a entrenar i es farà servir.
Gràcies a l'eina WandB que recomanen els propis autors del YOLOv5, es poden aconseguir i guardar tota la informació dels experiments fets amb la xarxa.

Inicialment, es fa amb una configuració de YOLOv5l amb imatges de 640x640, amb una precisió gran però amb una gran capacitat de còmput necessària. Els resultats no van ser molt satisfactoris, ja que les diferències entre les peces d'art amb alt grau de detall no tenien un bon resultat. Després de provar diferents solucions, la que millors resultats va donar va ser la d'augmentar la resolució de les imatges a 1280x1280.

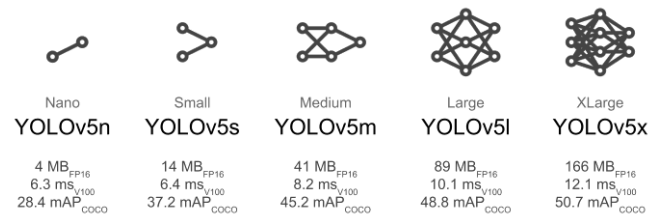


Fig. 4. Diferents models de YOLOv5

Per aquest motiu es va haver de baixar la complexitat de la xarxa a una YOLOv5s6 (el 6 del final denota una arquitectura especial per a imatges amb dimensió 1280x1280).

A més, el YOLO té una funcionalitat anomenada "*Hyperparameter Evolution*" que serveix per optimitzar els hiperparàmetres del model fent petites execucions amb petites modificacions automàtiques als hiperparàmetres del model per trobar la millor configuració pel problema específic.

En la taula 2 es poden observar els resultats de les diferents configuracions.
Per la tasca d'integrar el model de YOLOv5 entrenat a l'entorn de Flutter es van fer servir els plugins de Flutter *tflite* [28] i *tflite_flutter* [29]. Aquest procés no va donar resultats perquè els tensors d'input de la xarxa de YOLOv5 no consten dintre dels compatibles amb aquests dos plugins. Aquest fet l'originava el procés d'extracció del model YOLOv5 al format .tflite.

Model	YOLOv5m	YOLOv5l	YOLOv5s6
Dimensions	640x640	640x640	1280x1280
Epochs	1000	800	1000
Temps	13m45s	48m51s	12h49m
mAP_0.5	0.75	0.87	0.95
mAP_0.5:0.95	0.56	0.75	0.87
Recall	0.64	0.72	0.82
Precisió	0.87	0.82	0.91

Taula 2. Resultats de les configuracions de YOLOv5

6.3 Creació de dataset i model d'Image Classification

A conseqüència de la incompatibilitat del model exportat de YOLOv5 amb l'entorn de Flutter, es van plantejar les alternatives d'entrenar un model YOLO anterior que fos compatible amb els plugins o canviar el tipus de model a un d'Image Classification. Tampoc van funcionar xarxes com MobileNetV2 ni EfficientDet_Lite en Object Detection.
Com que l'usuari final només veurà de quina obra d'art es tracta, la decisió final va ser canviar el model només per la part de desenvolupament local. El desenvolupament en cloud continuarà tenint com a model un YOLOv5.
El dataset consta de les mateixes imatges que s'utilitzaven en l'Object Detection, però en aquest cas en comptes d'etiquetar-les amb caixes, s'organitzen en directoris on cada directori és la classe (obra d'art) a la qual pertany. Per tant, acabarem tenint un directori general que contindrà un directori per a cada obra d'art que existeix.
El model entrenat és un MobileNetV2 per a Image Classification que rep una imatge i retorna la classe amb més confiança juntament amb el seu percentatge.
El resultat és bastant bo i es pot comprovar en la taula 3, comparant el model d'Object Detection amb el d'Image Classification.

Model	YOLOv5m	YOLOv5l	YOLOv5s6	MobileNetV2
Dimensions	640x640	640x640	1280x1280	224x224
Epochs	1000	800	1000	35
Temps	13m45s	48m51s	12h49m	5m
Precisió	0.87	0.82	0.91	0.87

Taula 3. Comparacions de resultats de YOLOv5 amb MobileNetV2. Cal recordar que com que un classificador només prediu la classe de la imatge no hi pot haver falsos negatius, llavors la recall no té sentit en aquesta comparació

6.4 Estat aplicació progrés I

L'estat del projecte al final del primer progrès consta de dos models (YOLO i MobileNet) entrenats per detectar quina obra és i si es falsa o no. A més, el model de MobileNet està perfectament integrat en l'entorn local de Flutter i és completament usable tant en emulador com en un telèfon mòbil real.

L'aplicació actual és una interfície molt bàsica que dona a l'usuari l'opció d'introduir una foto des de la galeria del telèfon o directament fent una foto amb la càmera. Posteriorment, l'aplicació retorna la imatge amb el nom de l'obra resultant i la confiança d'encert. Tot aquest procés es veu a la figura 5.
Un problema que presenta el model classificador actual és que només funciona amb imatges en format horitzontal. Si rep una imatge amb orientació vertical, l'eficàcia del model baixa dràsticament, tenint un 5-10% de precisió en

les seves prediccions.

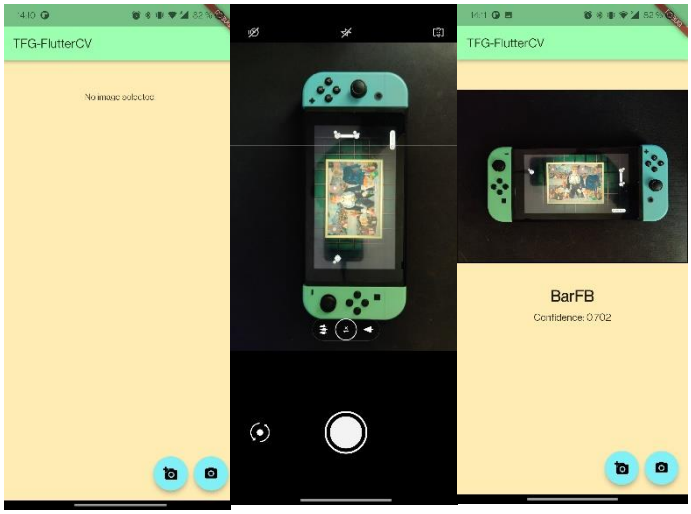


Fig. 5. Demostració d'ús de l'aplicació

La causa d'això és que el model ha estat entrenat amb prop de 800 imatges en format 4:3 amb 4000x3000 píxels (càmera de mòbil) i per introduir-les al model s'han redimensionat a factor d'escala 1:1. Això fa que les imatges s'estirin o s'encongeixin en direccions diferents si és una fotografia horitzontal o vertical.

A l'Apèndix 3 es poden trobar exemples de resultats de la fase de validació dels models, tant del YOLO com del MobileNet. Per a més detall en els experiments i els resultats, consultar el dossier del projecte.

7. Progrés del projecte II

Aquesta fase del projecte ha tingut com a nucli de desenvolupament mòduls d'OCR. A més, s'han millorat aspectes dels mòduls de Image Classification i s'ha desenvolupat l'API per poder utilitzar el model prèviament creat de YOLOv5 en un entorn cloud.

Pel que fa a la planificació inicial del projecte, no han hagut grans modificacions en els temps de cada una de les tasques. Per tal de tenir una millor perspectiva del projecte en el temps, es disposa de l'apèndix 4 on es mostra el diagrama de Gantt de tot el projecte fins l'estat actual.

7.1 Creació de l'API de YOLOv5

Per tal d'implementar la part Cloud del mòdul d'Object Detection, la primera aproximació va ser investigar diferents alternatives Cloud gratuïtes o amb períodes de prova per poder allotjar un model entrenat i que l'aplicació només s'encarregués de fer crides a aquesta API Cloud.

L'aproximació més propera a aquesta idea era utilitzar la plataforma de Google Cloud juntament amb Firebase. Com que Flutter i Firebase [30] són de Google, existeix molta documentació i el procés és relativament senzill. El principal problema és que en aquest servei el model es descarrega en el telèfon mòbil i es allà on es fan les inferències. L'aspecte Cloud només ajuda en no haver-se de descarregar una versió nova de l'aplicació a qualsevol canvi al model, si no que es descarrega en local la nova versió.

Com que aquest no era l'objectiu del projecte, el que es va decidir va ser emular una experiència Cloud creant una API amb l'ajuda del paquet Flask [31] de Python. El funcionament és molt simple: Un script de Python funciona com a endpoint de l'API la qual rep peticions HTTP des del telèfon mòbil amb una fotografia que pot sortir tant de la càmera com de la galeria. El programa a l'ordinador fa la inferència sobre la imatge i retorna el resultat (classe predita i confidence) en format JSON que després l'app de Flutter el descodifica i ho mostra per pantalla.

7.2 Mòdul OCR local

Per la implementació d'aquest mòdul es van fer tests amb 2 tecnologies diferents: Tesseract [32] i EasyOCR [33]. També estava l'opció de Keras-OCR [34], però es va descartar perquè no hi havia un mètode viable per incloure'l directament a Flutter, com si que hi ha amb els altres 2.

En l'entorn del projecte, les proves van afavorir Tesseract quan aquest es processava amb els recursos del telèfon. Aquest mètode funciona millor quan el text és desordenat i amb un reconeixement de text, a més de treure molt més rendiment de la CPU quan aquesta és limitada. En canvi, EasyOCR té millors resultats amb texts ordenats i estructurats com pot ser una factura i extreu millors resultats reconeixent números. Es pot veure aquesta comparació en profunditat en aquest article [35].

El programa implementat directament a Flutter és capaç d'identificar tots els texts que apareixen en una imatge determinada. D'aquest text processat, s'extreu la categoria d'animal que hi ha. En futures actualitzacions del mòdul es mostrarà informació important amb aquest output.

7.3 Mòdul OCR Cloud i dataset de pel reconeixement de zones de text

En aquesta implementació s'ha optat per entrenar un model personalitzat amb l'ajuda del YOLOv5 entrenat prèviament. D'aquesta manera, la part de reconeixement de text del model d'OCR ve determinat per un detector de bombolles de text del joc. En la figura 6 es veu un exemple d'aquestes bombolles amb la corresponent detecció.

Fig. 6. Detecció de bombolles de text

Per tal d'entrenar el model de detecció de text, es s'ha creat un dataset de fotografies preses amb un telèfon

mòbil i amb diferent grau d'inclinació, llum i distància. Aquest dataset ha estat etiquetat i allotjat a la plataforma Roboflow, tal com es va fer en el primer YOLO que es va entrenar.
Quan la zona on hi ha text ha estat reconeguda, la part d'OCR s'aplica sobre ella. En aquest cas s'ha escollit EasyOCR pel fet que té un rendiment superior en cas d'utilitzar la potencia d'una GPU.
D'igual manera que l'Image Recognition, la versió en Cloud envia la imatge des del telèfon mòbil a l'ordinador a on s'està executant el procés d'escolta per direcció IP. Quan rep la petició HTTP, processa la imatge amb els mètodes que s'han descrit. La resposta del programa és un JSON amb el resultat de la predicció i el grau de confiança.

7.4 Detecció d'espècie des de l'enciclopèdia

L'OCR segueix sent el mètode principal en aquesta funcionalitat. La primera aproximació va ser crear 2 versions com es porta fent en tot el projecte.

L'inconvenient que es va trobar va ser que el detector de bombolles de text no funcionava en aquesta pantalla, ja que la forma i el color del requadre no era el mateix. Depenent de la llum i la fotografia, era capaç de reconèixer la zona en un 40-50% de confidence, però en molts casos no ho detectava.

Per aquesta raó i per la falta de temps es va optar per només fer servir 1 proposta en aquest mòdul.
S'utilitza el mateix paquet que es fa servir per l'OCR local, EasyOCR. El funcionament és molt senzill: busca en el text que surt per pantalla l'espècie de l'animal i si surt el segell del museu. Amb tota aquesta informació, l'aplicació busca si l'usuari té marcat com a vist aquest animal en concret i si el té en el museu. A més, l'aplicació canvia de pàgina per mostrar la informació sobre l'animal que s'ha enregistrat.

7.5 Estat aplicació progrés II

L'aplicació ha canviat totalment la seva interfície i s'ha incorporat la pantalla d'inventari per a cada tipus d'espècie (peix, insecte, criatures del fons marí i art). Així doncs, es disposa de pàgines diferents que es desplacen amb un *Bottom Navigator Bar*.

Actualment l'única opció disponible en polsar el botó de càmera o galeria és el de reconeixement d'art. Aquesta rep la imatge que es selecciona i et mostra un diàleg emergent amb la predicció demanant una confirmació. En cas afirmatiu, l'aplicació es desplaça a la pantalla de l'obra d'art en qüestió.
En la següent entrega, s'espera tenir desenvolupada una bona pàgina d'informació i tots els mòduls d'OCR i Object Recognition implementats, tant en la versió cloud com la versió local.

8. Resultats

Apart dels resultats dels mòduls que ja han estat presentats, tenim com a resultat final del projecte dues aplicacions que si bé les seves funcions són idèntiques, tenen petites diferències. En la taula 4 es poden veure les característiques de cada una i també les seves diferències.

	Local	Cloud
Reconeixement d'art	✓	✓
Localitza l'art i pot reconèixer més d'1		✓
Predicció d'art com a categoria	✓	
OCR en frases de captura	✓	✓
OCR en pàgina d'enciclopèdia	✓	✓
Detecta a la pàgina d'enciclopèdia si l'espècie esta al museu	✓	
Requereix connexió a internet		✓

Taula 4. Comparacions de funcionalitats per versió d'aplicació

9. Següents passos

Aquests són els objectius que es treballaran en les següents entregues:

- Implementar tots els mòduls que s'han desenvolupat en les versions Cloud i local de les aplicacions.
- Millorar l'informe de cara a l'entrega final: Agrupar els resultats de totes les fases del projecte en una sola secció de resultats, canviar l'estructura de progressos per una que avarqui tot el projecte i incloure resum del projecte i abstract.
- Acabar d'elaborar el dossier del projecte.
- Crear el pòster del projecte.

10. Conclusions Provisionals

Estant en una fase tan avançada del projecte ja es poden treure conclusions dels resultats obtinguts fins ara.

En general, es pot concloure que els objectius principals s'han complert amb solvència. S'ha pogut comprovar que la implementació de mòduls de Visió per Computació en un sistema amb menys recursos com un telèfon mòbil és viable. Encara falta camí per recórrer per fer-ho d'una manera nativa però ja hi ha un avenç important.

També s'ha pogut explorar en un possible output o interfícies interactives que podrien tenir els programes que desenvolupem en la carrera d'Enginyeria Informàtica en el mercat real, on l'usuari final és un client que pugui utilitzar totes les funcions implementades sense tenir idea de que hi ha per darrera o tenir cap coneixement del camp.

Pel que fa a les dues versions de l'aplicació, totes dues funcionen d'una manera satisfactòria complint amb els objectius inicials. Pel que fa a les diferències entre les aplicacions, en l'estat actual i les implementacions del projecte, es troba el sentit de la comparació entre local i Cloud, veient millors rendiments en la majoria de funcions Cloud ja que els recursos de còmput són molt més potents. Si l'aplicació sortís al mercat amb les funcions actuals, es faria un mix entre les característiques locals i les Cloud, escollint totes les funcions de Cloud menys l'OCR de l'enciclopèdia, ja que en la versió en local s'aconseguien millors resultats i funcionalitats addicionals.
Encara que falti una última entrega abans d'acabar el projecte, es pot considerar el treball com un èxit.

BIBLIOGRAFIA

[1] "Animal Crossing: New Horizons - Wikipedia". Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Animal_Crossing:_New_Horizons (accedit el 17 de febrer de 2022).

[2] "Unit Sales of Animal Crossing: New Horizons worldwide as of December 2021". Statista. https://www.statista.com/statistics/1112631/animal-crossing-new-horizons-sales/ (accedit el 4 de març de 2022).

[3] "Animal Crossing awards and nominations". IMDb https://www.imdb.com/title/tt10476972/awards/ (accedit el 4 de març de 2022).

[4] "Technical Specs - Nintendo Switch™ - System hardware, console specs - Nintendo - Official Site". Nintendo. https://www.nintendo.com/switch/tech-specs/#switch-section (accedit el 18 de febrer de 2022).

[5] Chien-Yao Wang, Alexey Bochkovskiy, Hong-Yuan Mark Liao, "YOLOv4: : Optimal speed and accuracy of object detection" [Online]. Disponible: https://arxiv.org/abs/2004.10934

[6] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, "SSD: Single Shot MultiBox Detector" [Online]. Disponible: https://arxiv.org/abs/1512.02325

[7] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár, "Focal Loss for Dense Object Detection" [Online]. Disponible: https://arxiv.org/abs/1708.02002

[8] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" [Online]. Disponible: https://arxiv.org/abs/1506.01497

[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick, "Mask R-CNN" [Online]. Disponible: https://arxiv.org/abs/1703.06870

[10] Zhaowei Cai, Nuno Vasconcelos, "Cascade R-CNN: Delving into High Quality Object Detection" [Online]. Disponible: https://arxiv.org/abs/1712.00726

[11] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, Baining Guo, "Swin Transformer V2: Scaling Up Capacity and Resolution" [Online]. Disponible: https://arxiv.org/abs/2111.09883

[12] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso and Antonio Torralba, "ADE20K Dataset". https://groups.csail.mit.edu/vision/datasets/ADE20K/ (accedit el 25 de febrer de 2022).

[13] Tsung-Yi Lin, Genevieve Patterson, Matteo R. Ronchi, Yin Cui, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, Larry Zitnick, Piotr Dollár, "COCO Dataset". https://cocodataset.org/#home (accedit el 25 de febrer de 2022).

[14] Li Fei-Fei, Jia Deng, Olga Russakovsky, Alex Berg, Kai Li, "ImageNet Dataset". https://www.image-net.org/index.php (accedit el 25 de febrer de 2022).

[15] Mikey Taylor, "Computer Vision with Convolutional Neural Networks". https://medium.com/swlh/computer-vision-with-convolutional-neural-networks-22f06360cac9 (accedit el 27 de febrer de 2022).

[16] Grace Karimi, "Introduction to YOLO Algorithm for Object Detection", https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/ (accedit el 27 de febrer de 2022).

[17] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, Jiajun Liang, "EAST: An Efficient and Accurate Scene Text Detector" [Online]. Disponible: https://arxiv.org/abs/1704.03155v2

[18] Chandra Churh Chatterjee, "An Approach Towards Convolutional Recurrent Neural Network", https://towardsdatascience.com/an-approach-towards-convolutional-recurrent-neural-networks-a2e6ce722b19 (accedit el 27 de febrer de 2022).

[19] Javinpaul, "Top 5 Programming languages for Mobile App Development in 2022", Medium. https://medium.com/javarevisited/top-5-programming-languages-for-mobile-app-development-in-2021-19a1778195b8 (accedit el 27 de febrer de 2022).

[20] "Flutter Documentation". Flutter - Build apps for any screen. https://flutter.dev/learn (accedit el 17 de febrer de 2022).

[21] "What is Scrum?" Scrum.org. https://www.scrum.org/resources/what-is-scrum (accedit el 20 de febrer de 2022).

[22] "Jira | Issue & Project Tracking Software | Atlassian". Atlassian. https://www.atlassian.com/software/jira (accedit el 20 de febrer de 2022).

[23] "Notion - One workspace. Every team". Notion. https://www.notion.so/product (accedit el 20 de febrer de 2022).

[24] "Build software better, together". GitHub. https://github.com/about (accedit el 20 de febrer de 2022).

[25] "Meet Toggl. Three producs; One mission". https://toggl.com (accedit el 20 de febrer de 2022).

[26] "Overview – Roboflow". Roboflow. https://docs.roboflow.com (accedit el 11 de març de 2022).

[27] "Weights & Biases". WandB. https://docs.wandb.ai (accedit el 14 de març de 2022).

[28] Tflite Package. https://pub.dev/packages/tflite (accedit el 14 de març de 2022).

[29] Tflite_flutter Package. https://pub.dev/packages/tflite_flutter (accedit el 17 de març).

[30] Firebase. https://firebase.google.com (accedit el 15 d'abril de 2022).

[31] Flask User Guide. https://flask.palletsprojects.com/en/2.1.x/#user-s-guide (accedit el 15 d'abril de 2022).

[32] "Tesseract OCR: Text localization and detection". Tesseract. https://pyimagesearch.com/2020/05/25/tesseract-ocr-text-localization-and-detection/ (accedit el 20 d'abril de 2022).

[33] EasyOCR. https://github.com/JaidedAI/EasyOCR (accedit el 20 d'abril de 2022).

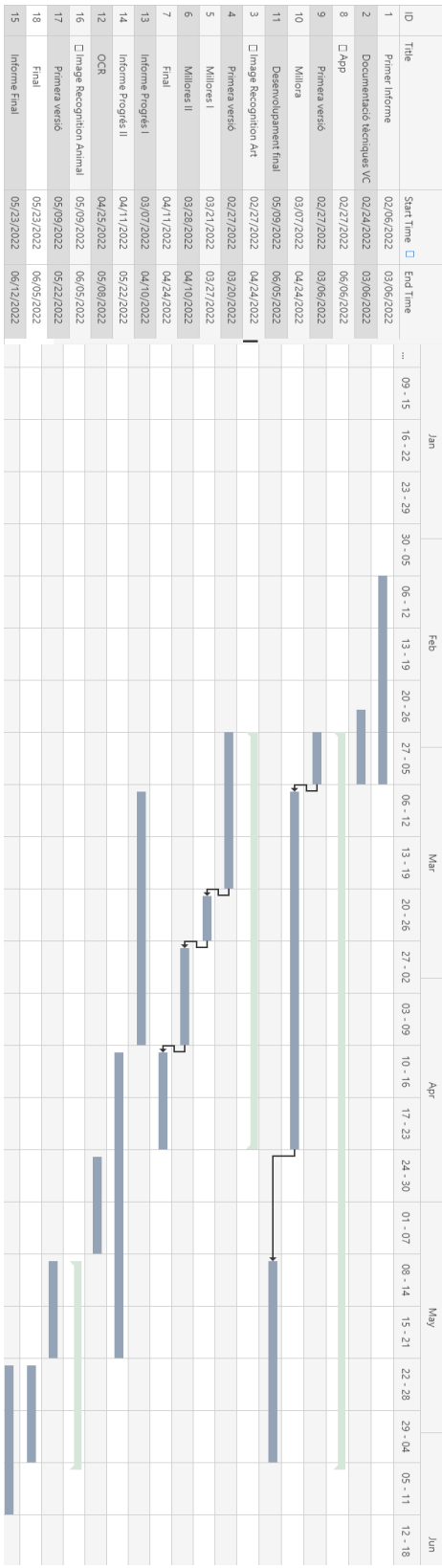
[34] Keras-OCR. https://github.com/faustomorales/keras-ocr (accedit el 21 d'abril de 2022).

[35] "OCR engine Comparison – Tesseract vs EasyOCR". https://medium.com/swlh/ocr-engine-comparison-tesseract-vs-easyocr-729be893d3ae (accedit el 21 d'abnril de 2022).

[36]

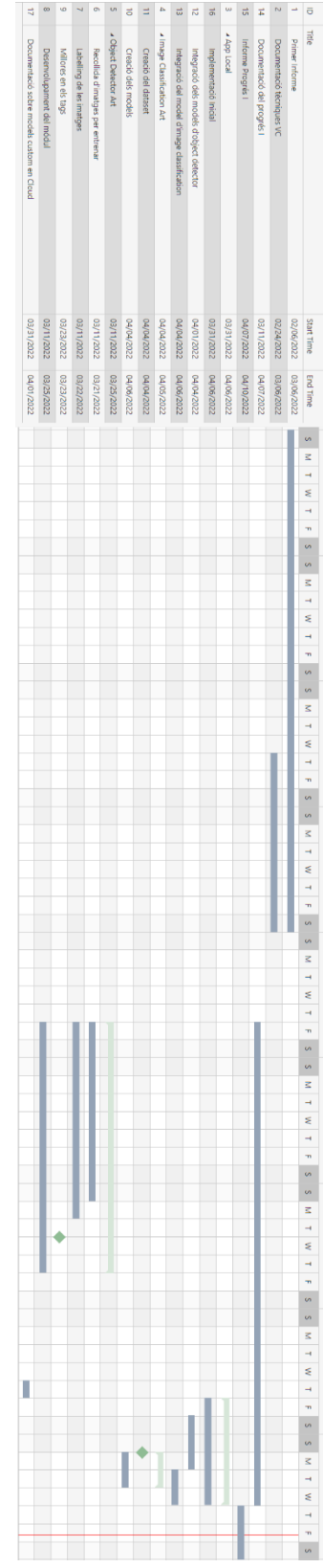
APÈNDIX

A1. DIAGRAMA DE GANTT INICIAL



ID	Title	Start Time	End Time
1	Primer Informe	02/06/2022	03/06/2022
2	Documentació tècniques VC	02/24/2022	03/06/2022
3	App	02/27/2022	06/06/2022
9	Primera versió	02/27/2022	03/06/2022
10	Millora	03/07/2022	04/24/2022
11	Desenvolupament final	05/09/2022	06/05/2022
3	Image Recognition Art	02/27/2022	04/24/2022
4	Primera versió	02/27/2022	03/20/2022
5	Millores I	03/21/2022	03/27/2022
6	Millores II	03/28/2022	04/10/2022
7	Final	04/11/2022	04/24/2022
13	Informe Progrés I	03/07/2022	04/10/2022
14	Informe Progrés II	04/11/2022	05/22/2022
12	OCR	04/25/2022	05/08/2022
16	Image Recognition Animal	05/09/2022	06/05/2022
17	Primera versió	05/09/2022	05/22/2022
18	Final	05/23/2022	06/05/2022
15	Informe Final	05/23/2022	06/12/2022

A2. DIAGRAMA DE GANTT PROGRÉS I



ID	Title	Start Time	End Time
1	Primer informe	02/06/2022	03/06/2022
2	Documentació tècniques VC	02/24/2022	03/06/2022
14	Documentació del progrés I	03/11/2022	04/07/2022
15	Informe Progrés I	04/07/2022	04/10/2022
3	App Local	03/31/2022	04/06/2022
16	Implementació inicial	03/31/2022	04/06/2022
12	Integració dels models d'object detector	04/01/2022	04/04/2022
13	Integració del model d'image classification	04/04/2022	04/06/2022
4	Image Classification Art	04/04/2022	04/05/2022
11	Creació del dataset	04/04/2022	04/04/2022
10	Creació dels models	04/04/2022	04/06/2022
5	Object Detector Art	03/11/2022	03/25/2022
6	Recollida d'imatges per entrenar	03/11/2022	03/21/2022
7	Labelling de les imatges	03/11/2022	03/22/2022
9	Millores en els tags	03/23/2022	03/23/2022
8	Desenvolupament del mòdul	03/11/2022	03/25/2022
17	Documentació sobre models custom en Cloud	03/31/2022	04/01/2022

A3. Validacions dels Models

Ground truth dataset Object Detection

Predicció model YOLO

Predicció model MobileNet

A4. Diagrama de Gantt progrés II