Detecció d'imatges modificades aplicant *Machine Learning*

Pol Ortiz Andreu

Resum– En aquest treball, es plantejarà la construcció i estudi d'una plataforma que permeti realitzar l'anàlisi d'una nova metodologia per a la detecció d'imatges modificades, basada en la combinació de l'estrategia *Divide And Conquer*, utilitzan xarxes neuronals i les característiques de les imatges.

Paraules clau– *Machine Learning*, xarxa neuronal experta, *DenseNet 169*, *Cloud*, Plataforma, *Big Data*, processament de dades, característiques de les imatges.

Abstract– In this project, the construction and study of a platform will be proposed to allow the analysis of a new methodology for the detection of modified images, based on the combination of the emph Divide And Conquer strategy, using neural networks trained with the properties of the image.

Keywords– *Machine Learning*, expert neural net, *DenseNet 169*, *Cloud*, Platform, *Big Data*, Data processing, image property

1 Introducció

AQUEST treball, realitzarà una anàlisi sobre l'efectivitat en la detecció d'imatges modificades a partir de la combinació de diferents xarxes neuronals entrenades només amb una característica de cada imatge.

Sorgeix aquest projecte a partir del fet que en la societat actual és usual l'acte de modificar imatges, tant sigui per motius estètics, amb aplicacions com Instagram o *Snapchat*, o per motius legals, com podria ser la pixelació de zones sensibles en una imatge, o per augmentar el rendiment en aplicacions com podrien ser les tècniques de súper mostreig utilitzant *machine learning* [1], etc.
Com s'ha pogut veure amb els exemples, el fet que sigui tan comú l'ús d'aquestes tècniques les ha portat a una evolució i perfecció del seu rendiment on, un ull inexpert podria confondre una imatge real amb una imatge retocada. Per tant, vista l'evolució en les tècniques de modificació, en aquest treball es busca replantejar les tècniques de detecció d'imatges manipulades.

El plantejament general del treball està enfocat tant en la construcció de la plataforma basada a l'algoritme *Divide and Conquer* [2] combinat amb les tecnologies de *Machine Learning*, com amb les anàlisis dels resultats obtinguts de les mateixes xarxes neuronals.

S'ha escollit l'algoritme *Divide and Conquer* perquè permet realitzar la següent interpretació de les xarxes neuronals:
Es posa el cas en què s'està mirant una mateixa imatge, per a detectar si ha estat modificada davant dels següents especialistes: un tècnic d'il·luminació, un escenògraf, un fotògraf i un editor d'imatges.
Cadascú es fixarà en les característiques que més coneix i podrà aportar una apreciació molt més precisa que si preguntéssim tot a una mateixa persona.
A més a més, s'incorporen les tecnologies de *Machine Learning* perquè ens permet implementar els rols que s'han esmentat anteriorment, on cada xarxa neuronal representarà a un dels experts anteriors i la seva especialització serà una de les característiques que extraurem de la imatge, de manera que cada xarxa neuronal aportarà la seva visió experta sobre la imatge i en conjunció de totes les xarxes expertes en treure'm un resultat que serà el veredicte.
D'aquest escenari sorgeixen les següents teories que desmentirem o verificarem en el treball: Una xarxa neuronal que presenta un bon rendiment en la classificació d'imatges es comportarà també de manera correcta només amb una característica de les imatges?, I diverses xarxes neuronals expertes donen millor rendiment que una sola xarxa neuronal?

• E-mail de contacte: polortizandreu@gmail.com
• Menció realitzada: Enginyeria de Computació
• Treball tutoritzat per: Jordi Serra Ruiz
• Curs 2018/19

2 OBJECTIUS

L'objectiu principal del projecte és realitzar una plataforma modular, basada en la filosofia de *Divide and Conquer*, que a través de les característiques de les imatges sigui capaç d'entrenar diferents xarxes neuronals expertes en cada una de les característiques, amb el fi de realitzar un estudi sobre la influència de l'extracció de característiques de les imatges per a la detecció d'imatges modificades.

En ser un objectiu principal tan complex, i amb l'objectiu d'obtenir una anàlisi qualitatiu, s'ha ramificat en dues branques, descrites a continuació.

2.1 Anàlisi

Aquest subobjectiu es focalitzarà en l'anàlisi de les característiques i al corresponent rendiment en xarxes neuronals expertes.

Els objectius principals que s'han establert per a l'anàlisi de les característiques són els següents:

1. **Anàlisi de la zona de confort en les xarxes neuronals**: Se suposa la següent premissa:
   Les característiques de les imatges són reinterpretacions o abstraccions de porcions, o no, d'una mateixa imatge, i les xarxes neuronals aconsegueixen extraure característiques de porcions d'imatges.
   Establertes aquestes concepcions, l'objectiu en qüestió focalitzarà en desmentir o verificar si una xarxa neuronal que dóna un bon resultat en la classificació d'imatges donarà també un bon resultat en la classificació de les característiques d'una imatge.

2. ***Divide & Conquer***: Aquest subobjectiu ve directament inspirat de l'algoritme *Random Forest*, on es buscarà principalment verificar l'efectivitat entre diferents xarxes neuronals comparades amb una única xarxa neuronal.

3. **Rendiment de diferents característiques**: A causa del fet que es realitza una xarxa neuronal experta per característica, en aquest subobjectiu es realitzarà un estudi sobre quines característiques han aportat un millor resultat en els experiments.

4. **Estudi de la informació**: Les característiques suposen eventualment o una transformació directa de la imatge, o una reducció en la informació resultant. Per exemple, unes característiques com podria ser un histograma, suposen una pèrdua en la informació molt més notòria, que no la que suposa la característica de la luminància.
   Per tant, en aquest objectiu es realitzarà una anàlisi sobre la repercussió que suposa la pèrdua d'informació respecte al comportament de les xarxes neuronals expertes.

2.2 Construcció

En aquest subobjectiu es focalitzarà en els objectius referents a la construcció de la plataforma per se:

1. **Big Data**: L'estructura de la plataforma que es construirà ha de permetre el processament de grans volums de dades. Recordar que el processament que es realitzarà per cada imatge és, extracció de diferents característiques, entrenament i predicció, si no s'estableix una estructura que permeti un emmagatzemament lògic i no s'estableix un control de les dades, es tindrà com a conseqüència el fet de duplicats de memòria i de repetició de processos d'alt cost computacional com és el càlcul de l'extracció de les característiques d'una imatge.
   Per tant, aquest objectiu focalitzarà en la creació d'una plataforma que consideri en tot moment els actes realitzats anteriorment per tal d'evitar duplicats tant de processos com de fitxers en memòria.

2. **Desacoblament**: El fet del coneixement dels tres àmbits que es toquen en aquest projecte: flux de dades, extracció de característiques i entrenament de xarxes neuronals expertes, pot resultar complex per a un futur desenvolupador i conseqüentment generar una distracció de l'objectiu principal.
   Per tant, aquest objectiu focalitzarà en facilitar la feina dels desenvolupadors que simplement vulguin modificar un dels àmbits a partir de realitzar una plataforma que desacobli al màxim aquests àmbits.

3. **Modularitat**: Com s'ha explicat anteriorment, el fet de realitzar una extracció, o un entrenament té un cost computacional i en temps elevat, i la majoria d'execucions simplement serviran per realitzar proves sobre el codi desenvolupat.
   Per tant, aquest objectiu focalitzarà en realitzar una plataforma que permeti de manera simple l'execució de la porció o l'àmbit que es desitgi.

4. **Base**: Aquest projecte realitza la tasca de construir la plataforma i de realitzar un primer estudi sobre les característiques de les imatges, però aquest objectiu focalitza en realitzar una plataforma genèrica, una base, que faciliti l'estudi de futurs projectes, que per tant proporcionin un coneixement més precís referent a les característiques de les imatges i les xarxes neuronals expertes.

3 ESTAT DE L'ART

Aquest projecte es basa en quatre pilars conceptuals: *Divide & Conquer*, característiques de les imatges, xarxes neuronals, i *Big Data*.
En l'estudi previ al treball, no s'han trobat treballs anteriors que combinin o presentin estructures similars a la plantejada al projecte, però tan bon punt separem els conceptes i els analitzem individualment ens trobem amb una quantitat de treballs que ens han permès establir la bases. Per tant, per a realitzar un informe sobre l'estat de l'art del projecte, es realitzarà de manera individual per a cada pilar.

3.1 *Divide & Conquer*

Com s'ha explicat anteriorment en aquest treball s'utilitzen una estratègia basada en la combinació de *Divide & Conquer* [2] i *Random Forest* [3]. D'aquestes dues

estratègies se n'extreu finalment la teoria que es tradueix en, convertir la feina que realitzava una xarxa neuronal entrenada amb imatges classificades en múltiples xarxes neuronals entrenades amb només una característica.

De manera que referent a *Divide & Conquer* estem dividint la tasca i la complexitat de la classificació en múltiples xarxes neuronals expertes, i referent a *random forest*, podem construir xarxes neuronals més especifiques i més simples, ja que el veredicte final el donarà la combinació de totes.

Finalment, per ser exactes amb la teoria de l'algoritme de *random forests*, s'haurien de realitzar multitud d'arbres (l'equivalent al treball són les xarxes neuronals) simples i aleatoris en estructura (profunditat) de manera que permetessin obtenir un veredicte correcte a partir de la combinació, però com el projecte focalitza en realitzar un estudi també sobre la pèrdua d'informació i el rendiment de les característiques, el fet de realitzar xarxes neuronals amb inicialització i construcció aleatòria, afegia complicacions i resultats dispersos per a l'anàlisi en qüestió.

3.2 Característiques de les imatges i *dataset*

Per a la realització de l'estudi de les característiques de les imatges, s'ha buscat primer quines característiques es podien extraure de les imatges i quin era el procediment d'extracció de les mateixes [4].
Aquest procés anava directament relacionat amb la recerca d'informació sobre projectes anteriors que combinessin aquests dos conceptes, amb l'objectiu de validar i contrastar els resultats, malauradament no s'han trobat treballs de qualitat suficient per a basar una anàlisi sobre aquests.

Aquestes característiques han de ser extretes d'un *dataset* d'imatges classificat per imatges verdaderes i falses, a més a més, ha de permetre saber sobre quines modificacions s'han aplicat a cada imatge per saber amb quina modificació cada xarxa neuronal obté un rendiment millor.
El *dataset* [5] compleix amb els requeriments establerts, ja que disposa d'una estructura amb les imatges originals i les imatges modificades estructurades en carpetes diferents.

Referent a la elecció del *dataset*, s'ha escollit el *dataset* [5], perquè realitza diferents modificacions a cada imatge i aquestes estan documentades, de moment que es podrà estudiar sobre quines modificacions cada xarxa neuronal experta presenta un millor resultat.
Les modificacions del dataset son:

- **orig**: Imatges originals, sense modificacions.
- **nul**: Copia directa d'una porció de la imatge.
- **lnoise**: Imatge amb l'afegit de soroll Gaussià.
- **rot**: Rotació de la imatge
- **scale**: Augmentació de parts de la imatge.
- **cmb**: Combinació de les modificacions anteriors.
- **multi_paste**: Porcions d'imatge copiades i enganxades multiples vegades.

3.3 Xarxes neuronals

Actualment, hi ha una gran multitud de treballs sobre xarxes neuronals i el rendiment d'aquestes. Un dels objectius principals era el de realitzar una anàlisi per a verificar o desmentir si una xarxa neuronal amb bon rendiment en classificació tindria un bon rendiment amb només les característiques. Per assolir aquest objectiu, es va realitzar una cerca referent a les xarxes neuronals que presenten millor rendiment respecte a la classificació d'imatges [6].
Finalment, es va concloure que s'utilitzaria la xarxa neuronal DenseNet169 [7], gràcies als bons resultats en la classificació d'imatges i a la fàcil implementació gràcies a la llibreria Keras [8].

3.4 *Big Data*

Referent als objectius sobre el processament i emmagatzemament de dades, es va realitzar un estudi sobre quines eren les eines amb les quals es construiria la plataforma. Els objectius principals en la cerca eren trobar una plataforma de ràpid desplegament i posada en marxa que permetis realitzar insercions i un control del flux de dades.
Finalment, s'ha optat per l'ús d'eines en el Cloud, ja que permeten obtenir de manera fàcil i rapida una implementació per al control tant d'emmagatzemament com de monitoratge del flux de dades i de les màquines virtuals, fet que ens permetrà conèixer en tot moment on s'estan ocasiona'n problemes en la plataforma.

4 METODOLOGIA

Per a la realització d'aquest projecte, és essencial el coneixent en tot moment de quina tasca s'està realitzant i el perquè d'aquesta tasca. Per a respondre a aquests dubtes, a continuació s'explicarà tant la metodologia que s'ha seguit pel desenvolupament de cada tasca com l'estructura i eines que s'han utilitzat per a la implementació del projecte.

4.1 Model incremental

El model de desenvolupament de projectes que més s'adequa a les necessitats és, el model incremental [9].
Aquest model de desenvolupament permet a cada iteració donar un resultat o una imatge del projecte, que s'anirà perfeccionant i complementant a mesura que passin les iteracions. Es va descartar l'ús d'una metodologia com el desenvolupament en cascada [10] perquè hi haurà modificacions en la implementació del model inicial de manera freqüent, a causa del fet que la majoria de les tasques comporten un estudi de les eines i una adaptació de l'estructura per a adaptar-les en la plataforma.
El model incremental requereix dues etapes principals: etapa d'inicialització, que es basarà en la construcció d'una primera estructura de la plataforma completa, on només hi haurà els mòduls definits, i una etapa d'iteració, on es realitzarà tant la implementació de cada mòdul com l'estudi i construcció de cada característica i la consegüent xarxa neuronal.
Les etapes del desenvolupament, es poden trobar en el Gannt de l'apèndix.

4.2 Estructura i eines

A continuació s'explicaran quines han estat les eines i recursos utilitzats per a la realització dels experiments i construcció de la plataforma.

Referent a l'entorn de programació, s'ha utilitzat el llenguatge de programació Python amb la complementació de les llibreries tant de modificació i tractament de matrius Numpy, com de tractament d'imatges OpenCV, com les llibreries referents a la creació de les xarxes neuronals, Tensorflow i Keras [8].
Aquesta decisió d'entorn ha estat escollida gràcies a la ràpida implementació que disposa el llenguatge de programació Python i a la facilitat en la cerca d'informació gràcies a l'activitat constant en les comunitats de desenvolupadors de les mateixes eines.

El desenvolupament d'una plataforma d'aquesta complexitat suposa directament la necessitat de l'ús d'una eina de control de versions, ja que es vol evitar en qualsevol etapa del desenvolupament la pèrdua o necessitat de re-implementació de parts programades.
Gràcies a la fàcil implementació i al coneixement previ de l'eina, s'ha utilitzat un repositori de GitHub, que permet, en cas de necessitat de *rollback*, retrocedir a *commits* anteriors i tenir un control constant de quins són els treballs modificats, per qui han estat modificats i per quin motiu han estat realitzats.

Una vegada establert l'entorn de programació, s'ha d'executar. Per a realitzar l'execució es necessitarà un clúster que disposi de com a mínim una GPU per a executar la xarxa neuronal *DenseNet169* [7] amb Tensorflow.
A causa de la dificultat en la configuració de les llibreries de CUDA necessaries per a executar TensorFlow i que utilitzi la GPU de manera correcta, s'ha optat per l'ús d'una instància de Google Compute Engine amb la imatge personalitzada de TensorFlow [11].
Per a l'emmagatzemament de les dades i control de les dades s'ha utilitzat un *Bucket* de Google Cloud Storage complementat amb una estructura de taules a BigQuery.

5 CONSTRUCCIÓ DE LA PLATAFORMA

En aquest apartat del treball, es realitzarà un estudi sobre la plataforma construïda, especificant quines han estat les millores i solucions per a assolir els objectius.
Durant aquest desenvolupament s'han generat dues versions de la mateixa plataforma, de manera que per entendre el resultat final, s'explicarà primer l'estructura de la primera iteració de la plataforma, els problemes trobats amb aquesta, i finalment s'explicarà com s'han solucionat aquests problemes amb la segona iteració de la plataforma.

5.1 Primera iteració: treballs locals

5.1.1 Procediment

Com s'ha definit en apartats anteriors del projecte i com es pot veure a la figura 1, per cada imatge se n'ha d'extreure les característiques, aquestes s'han de guardar els resultats, i finalment, entrenar per cada característica la corresponent xarxa neuronal experta, complint amb un disseny modular i genèric.
En la primera iteració el procés a seguir era el següent:

Primer de tot es realitza un procés de neteja i construcció de l'estructura del *dataset*, aquest procés permet establir unes bases de conveni per al tractament del *dataset*. Gràcies a aquest conveni en l'estructura, sabem que a nivells inferiors de cada capeta, tenim directament les imatges a les quals se'ls hi ha aplicat la modificació especificada en el nom de la carpeta, com es pot veure a la figura 12. Aquest procés només s'executarà una vegada per canvi de *dataset*.

Realitzada aquesta tasca, els procediments descrits a continuació s'executaran múltiples vegades, sigui per execucions d'experiments o per estudiar el comportament de les parts desenvolupades.
A més a més, i complint amb l'objectiu de modularitat cada part descrita a continuació pot ser executada per separat, només necessitarà el treball que especifiqui la informació de l'execució del bloc anterior. Per exemple, l'extractor de característiques necessita que se l'hi especifiqui el treball que conté la informació del *Dataset* generat.

Després de realitzar la construcció de l'estructura del *dataset*, s'ha de construir el *dataset* que es vol utilitzar, aquest acte es tradueix en l'activitat de seleccionar quines i quantes imatges desitgem enviar a l'extractor de característiques.

La construcció del *dataset* ha estat implementat per assolir l'objectiu de modularitat: s'ha de permetre i facilitar les tasques d'un futur desenvolupador de seleccionar només les imatges que desitgi. En conseqüència s'està assolint l'objectiu referent a l'optimització tant de temps com de cost computacional.
Per exemple, es té un *dataset* amb imatges modificades de classe *ModifA*, *ModifB* i *ModifC*, si només es volen extraure les imatges de *ModifB* perquè, per exemple, les imatges de *ModifA* ja han estat extretes i de moment les imatges amb *ModifC* no interessa extraure-les, simplement s'ha d'especificar al constructor del *dataset* que busquem extraure les característiques de *ModifB*.
El resultat d'aquesta execució, com es pot veure a la figura 2 es guardarà en un fitxer i el qual serà utilitzat per l'extractor de característiques.

Seguidament, l'extractor de característiques, rebrà com a informació dos conjunts d'informació: imatges que ha d'extraure i quines característiques ha d'extraure de cada imatge.
Gràcies a aquesta inicialització el disseny està assolint, també, l'objectiu de modularitat, ja que el desenvolupador final només ha d'especificar quines imatges desitja i quines característiques desitja extraure.
L'execució que realitzarà l'extractor és realitzar la tasca de: recórrer totes les imatges especificades en el *dataset*, i per cada imatge extreure les característiques que s'han especificat també.

En la construcció de la plataforma s'ha especificat l'objectiu principal de màxim desacoblament entre la implementació d'una característica i el funcionament de

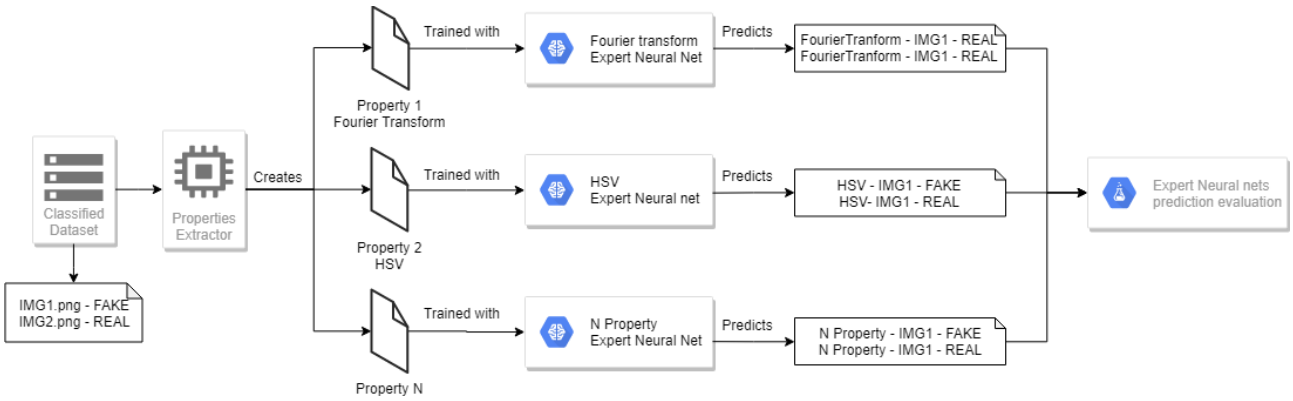

Fig. 1: Estructura genèrica del flux de dades de la plataforma.

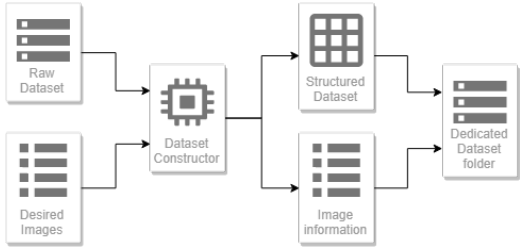

Fig. 2: Estructura del flux de dades en la construcció del *dataset* en la primera iteració.

l'extractor de característiques, per a aconseguir aquest objectiu s'ha establert un sistema de *call backs* que aïlla la implementació de l'extractor, de les característiques per se. La qualitat de l'aïllament aconseguit converteix la complexa tasca d'afegir un mòdul a un programa tancat a tres tasques simples:
Si un desenvolupador vol programar una nova característica que extragui, per exemple, la transformada de Fourier d'una imatge haurà de, primer definir una funció en un fitxer específic, com es pot veure en l'annex, on com a entrada ha de tenir una imatge i de sortida una matriu bi-dimensional, definir un objecte on s'especifica el nom de la característica i el *callback* a la funció desenvolupada i afegir-lo en un diccionari específic, finalment per provar la característica només ha d'executar la plataforma especificant el nom establert per a la característica.
Per darrere el que s'està realitzant és un diccionari d'objectes, amb la informació de cada característica, indexat per nom, amb el resultat que, quan s'especifiquen X noms de característiques, l'extractor recórrer el diccionari genèric i executa el *callback* especificat.
Les característiques extretes es guardaran cada una en fitxer i es generarà un fitxer amb les característiques extretes en aquesta execució.

Finalment, amb les xarxes neuronals es tindrà un procés similar a l'extractor de característiques, el qual s'explicarà amb més detall en l'apartat de xarxes neuronals. En línies generals, es generaran totes les xarxes neuronals i s'entrenaran amb les característiques que indiquem, quan hagin finalitzat l'entrenament, es guardarà l'estat de la xarxa neuronal i es realitzaran les prediccions de cada valor especificat en el *dataset*, sigui del conjunt de *train* o de *test*. Aquestes prediccions realitzades es guardaran en un fitxer.

5.1.2 Problemes

Com es pot veure en l'apartat anterior, per a cada execució es generen N treballs, en N carpetes i aquest fet genera ràpidament quatre problemes principals:

- **Complexitat**: El fet de gestionar estructures de carpetes complexes on per cada execució es generen fitxers, complica de manera significativa el procés tant de cerca com de control de les dades que es tenen.
  Per exemple, com s'ha explicat anteriorment per a cada extracció de característiques es genera un fitxer amb les dades extretes, el fet de tenir un fitxer per execució es controla posant com a nom del fitxer un timestamp i situant aquest fitxer en la carpeta de la característica extreta, de manera que, per exemple, en una execució de l'extracció de la característica *CarA* de la imatge *A*, s'ha de recórrer tots els fitxers de la carpeta *CarA* en cerca de la imatge *A*, per comprovar si s'ha extret anteriorment.

- **Memòria**: A causa del fet que sigui tan complex tenir un control en tot moment de les tasques realitzades prèviament, i d'haver de realitzar una cerca costosa per imatge, no es realitzava en la majoria d'execucions el control de duplicats. Òbviament, aquest fet ocasionava directament un problema de duplicats per execució.
  A més a més, i de manera independent, al problema de la complexitat en l'extracció de característiques es generen una quantitat de dades que pot ocasionar problemes d'emmagatzemament.
  Per exemple, si extraiem les característiques *A*, *B* i *C* que no presenten pèrdua d'informació, com pot ser *HSV*, per a 500 imatges de volum 50Mb, s'estan generant: 50Mb * 500 imatges * 3 característiques = 75 Gb en una sola execució.

- **Temps**: També ocasionat per la falta de control de duplicats, es repeteixen processos d'extracció de d'imatges, fet que causarà repetir tant el cost en temps com el cost computacional en l'extracció de X característiques per a Y imatges.

- **Explotació**: El fet de gestionar amb diferents arxius i de no disposar d'una base de dades, l'explotació dels resultats obtinguts s'hauria de fer manual, i vista la

complexitat en la estructura de les carpetes suposa un cost tant en temps de desenvolupament com d'execució molt elevat, per cada gràfica a extraure.

Una vegada la primera iteració de la plataforma va estar construida i en funcionament, es va estudiar si assolia o no els objectius plantejats. Vistes les complicacions i problemes obtinguts, es va realitzar un estudi per a disminuir els problemes presents i complir amb els objectius inicials.

5.2 Segona iteració: Cloud

En aquesta segona iteració de la plataforma s'ha establert com a objectiu resoldre els problemes de la primera iteració de la plataforma i aconseguir complir amb els objectius principals establerts sobre la plataforma, explicats anteriorment.

Com es pot veure en la figura 3, el flux de dades i el procediment que seguiran les imatges és molt similar al disseny establert en la iteració 1, però, per assolir els objectius, en la segona iteració s'ha substituït tot el sistema de fitxers i emmagatzemament local per tecnologies Cloud.

Com es pot veure en la figura 3, la gestió del control d'accions realitzades prèviament, ja no està realitzat per un sistema de fitxers, sinó que ara s'utilitza una base de dades, BigQuery. Per a realitzar el control de les accions realitzades s'ha creat a BigQuery un *dataset* amb 3 taules:

1. **image_dataset**: És una taula dedicada a guardar la informació del *dataset*, cada entrada representa una imatge del *dataset*.
   Gràcies a tenir aquesta taula, independentment d'on localitzem les imatges, s'obté un control absolut de les imatges amb una simple consulta, ja no s'han de consultar les anteriors execucions per buscar una imatge concreta.

2. **image_properties**: És una taula dedicada a guardar la informació de les característiques extretes, cada fila representa una característica extreta i conté informació de la mateixa com per exemple quin ha estat el mètode d'extracció, on està guardada, i les característiques de la imatge de la qual prové.
   De similar manera que en el punt anterior, gràcies al coneixement a través d'una simple consulta a la base de dades el control de duplicats és molt més simple i viable en execució.

3. **predictions**: És una taula dedicada a guardar la informació referent a les prediccions realitzades per les xarxes neuronals expertes, cada inserció a la taula fa referència a la informació relacionada a la predicció d'una característica d'una xarxa neuronal.
   Gràcies a tenir emmagatzemat en una base de dades les prediccions, aquestes les podem explotar amb eines de Business Inteligence com per exemple Data Studio.

A més a més, com es pot veure en el diagrama 3, s'utilitzen la tecnologia de Google Cloud Storage, amb aquest afegit, s'obtenia l'avantatge de tenir el *dataset* distribuït i plantejava també la solució a l'emmagatzematge de les característiques, permeten així l'execució de la plataforma en computadors que no disposin de tanta memòria.

Així doncs, gràcies a les millores realitzades en afegir aquestes tecnologies, s'han aconseguit assolir els objectius establerts i minimitzar els problemes ocasionats en la primera iteració de la plataforma.

6 XARXES NEURONALS EXPERTES

En aquest apartat s'explicarà de manera detallada tant la construcció de les xarxes neuronals com les eleccions de les característiques que serviran com a input d'aquestes.

6.1 Disseny

Primer de tot s'ha d'explicar com es generen les xarxes neuronals. Complint amb els objectius establerts de disseny, s'ha realitzat un disseny de la plataforma que aconsegueixi el màxim desacoblament entre el desenvolupador i la implementació de la plataforma.

Les diferents xarxes neuronals han estat construïdes, com es pot veure en l'apartat de *Expert neural net build* a la figura 3, a partir d'una *factoria* de xarxes neuronals.
La *factoria* de xarxes neuronals necessita que se li especifiquin dos valors: les xarxes neuronals que es volen executar, així es compleix amb l'objectiu de modularitat, i el *dataset* que es vol utilitzar, així també es compleix amb l'objectiu de control de les dades.
Una vegada inicialitzat, la *factoria* retornarà una llista amb xarxes neuronals expertes, amb les quals s'hauran de realitzar les accions necessàries com construint la xarxa neuronal, entrenar-la, ... Amb l'objectiu de poder realitzar els experiments.

Referent a la part de la construcció d'una xarxa neuronal experta, per aconseguir els objectius referents al desacoblament, s'ha creat una classe abstracta, *NeuralNetAbstract*, que tractarà gran part de la interacció amb la llibreria de Tensorflow.
Gràcies a aquesta abstracció s'aconsegueix estalviar temps al futur desenvolupador, distanciant les tasques que són més de coneixement d'una eina i les seves peculiaritats, que d'estudi del comportament.

Aquesta classe abstracta implementa els mètodes bàsics d'una xarxa neuronal:

- **Constructor**: En aquest mètode s'inicialitzen els valors i propietats de la classe.

- ***set_train_test***: En aquesta funció es realitza la partició del *dataset* en els dos conjunts de dades *train* i *test*, amb l'objectiu de validar el sistema.

- ***build_model***: En aquesta funció es realitza la partició en *train* i *test* amb l'objectiu de poder fer tant la definició com la construcció, utilitzant el mètode *compile* de Tensorflow, de la xarxa neuronal experta.

- ***transform_path_dataset2nn_input***: Aquesta funció no està implementada per la classe abstracte, perquè s'ha de sobreescriure per la classe fill. L'objectiu d'aquesta funció és, convertir la informació de cada característica a una matriu que servirà com a entrada per a la xarxa neuronal experta.

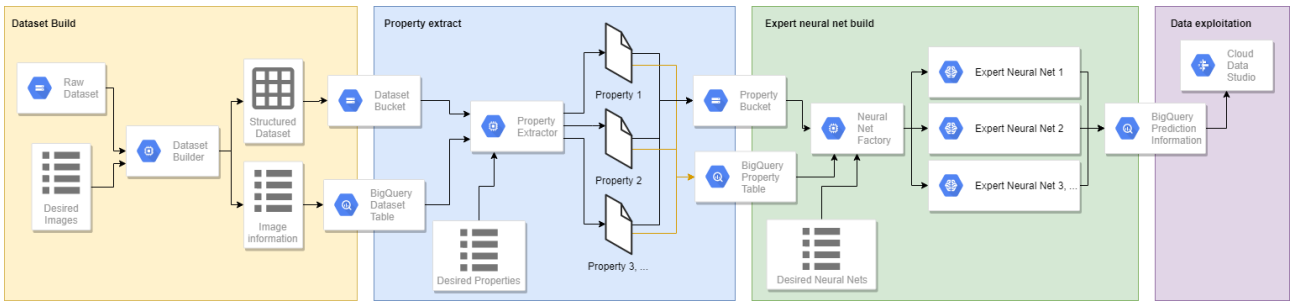

Fig. 3: Estructura final del flux de dades de la segona iteració de la plataforma.

- ***train***: En aquesta funció es realitza la funció *fit* de Tensorflow, per la qual s'entrena la xarxa neuronal.

- ***predict & save results*** En aquests dos mètodes es realitza la tasca de primer, recórrer tot el *dataset* i donar un veredicte, que es guardarà a BigQuery, i realitzar la tasca d'avaluar el sistema comparant les prediccions amb els valors del subconjunt de dades *test*.

Aquests mètodes poden ser fàcilment sobreescrits gràcies al fet que presenten una baixa dependència entre els mètodes de la classe.

Com s'ha explicat, totes les funcions estan implementades excepte una, *transform_path_dataset2nn_input*.
En aquesta funció el desenvolupador només ha d'implementar l'acte de convertir un *dataset* que conté la informació de les característiques en l'*input* final (en forma de matriu) per a la xarxa neuronal.
Per exemple, com es pot veure a la figura 14, el desenvolupador té com a objectiu crear una xarxa neuronal experta a partir de la característica de la transformada de Fourier. Per a realitzar aquesta tasca, només ha de crear una classe que hereti de la classe neuronal abstracta i implementar el métode *transform_path_dataset2nn_input*. Per assolir amb l'objectiu que la plataforma sigui genèrica, si un usuari vol redefinir el model de la xarxa neuronal o la metodologia per dividir en *train* i *test* el *dataset*, simplement haurà de sobreescriure els mètodes de la classe pare.

Finalment, i com s'ha explicat anteriorment, s'ha escollit la xarxa neuronal DenseNet169.
Per als experiments que s'han realitzat, per a obtenir uns resultats idèntics en totes les execucions, en tots els factors aleatoris, com pot ser la divisió en *train* i *test*, la inicialització dels pesos de la xarxa neuronal, ... Estan controlats a través d'una *seed*, que estableix un valor el qual farà que tots els valors aleatoris generats a la xarxa neuronal siguin idèntics.
La configuració de la xarxa neuronal es pot veure a la taula 1 i en els experiments realitzats, només es canviarà tant l'input com els *epochs* (nombre d'iteracions que realitza la xarxa neuronal sobre el dataset, abans de finalitzar l'entrenament).

6.2 Característiques escollides

En aquest apartat, s'explicaran quines han estat les característiques extretes i el perquè d'aquestes.

La raó de l'elecció de les característiques es basa en dos factors, que referencien directament als objectius d'estudi principal: Volem saber quin és el comportament d'una xarxa neuronal només amb una abstracció d'una imatge, i també volem saber quina és l'afectació de la pèrdua d'informació.

- *Original*: Aquesta característica que es veurà més endavant en gràfiques, no és una característica en si, sinó que és la imatge amb RGB. Gràcies a tenir aquesta característica, es pot realitzar una comparació directa amb els resultats obtinguts de la xarxa neuronal expertes entrenades amb altres característiques.

- *Harris Detector*: Aquesta característica és l'extracció de les cantonades en una imatge [12].
  S'ha escollit aquesta característica amb l'objectiu d'estudiar el comportament de la xarxa neuronal amb una petita pèrdua d'informació al convertir-se d'una imatge RGB a blanc i negre, però realitzant una abstracció de la imatge completament diferent.

- *Fourier Transform*: Aquesta característica és la transformació d'una imatge a l'espai de les freqüències [13].
  S'ha escollit aquesta característica amb el mateix objectiu que la característica *Harris Detector*, però amb un afegit, gràcies a tenir dues característiques amb abstraccions, es pot estudiar a través de comparacions quina de les dues característiques és més eficaç.

- *HSV & YUV*: Aquestes dues característiques són transformacions de l'espai de color de la imatge.
  S'han escollit aquestes característiques amb l'objectiu d'analitzar, sí, amb 0 pèrdues d'informació, només canviant l'espai de color, les xarxes neuronals responen de manera dispar.

- *Histogram*: Aquesta característica és l'extracció de l'histograma d'una imatge.
  S'ha escollit aquesta característica amb l'objectiu de comprovar si amb una pèrdua notòria d'informació en una característica, la xarxa neuronal experta és capaç de donar resultats convenients.

7 Resultats

A continuació s'explicarà en detall quins han estat els experiments realitzats amb les xarxes neuronals expertes, quins eren els valors esperats, i quina és l'explicació del comportament obtingut. Per a realitzar aquesta anàlisi s'ha

utilitzat l'eina d'explotació Data Studio, ja que, gràcies a tenir totes les dades de les prediccions guardades a BigQuery, l'explotació d'aquestes dades és més ràpida i simple.

L'estructura d'aquesta anàlisi es realitzarà a través de dos punts de vista diferents: Una anàlisi del comportament de cada xarxa neuronal expertes, i un segon anàlisi que farà referència al comportament de totes les xarxes neuronals expertes donant un veredicte.

Com es pot veure a la taula 1, tots els experiments han estat realitzats amb la mateixa configuració però amb diferents nombres d'*epochs*.
L'elecció de realitzar diferent nombre d'*epochs* és degut al fet que es vol fer un estudi del comportament d'una xarxa neuronal en vers a la informació de l'*input* donat, amb aquests tres números (i pel volum de dades amb el que s'està treballant), posem en tres situacions una xarxa neuronal: Poc entrenament (5 *epochs*), on es podrà veure quines són les característiques que donen resultats correctes tot hi el baix entrenament, entrenament mitjà (40 *epochs*), i sobreentrenament (100 *epochs*), gràcies a tenir aquest sobreentrenament, podrem avaluar quines xarxes neuronals s'estanquen en l'entrenament i no aprofiten els *epochs* extres cedits.

7.1 Anàlisi experiments xarxes neuronals expertes

En aquest apartat del treball, s'explicaran de manera extensiva els resultats obtinguts de cada xarxa neuronal experta.

7.1.1 Original

En aquesta xarxa neuronal experta, simplement observant la figura 4 podem saber quin es el comportament que presenta.
S'observa que té un *accuracy* creixent, fet que indicaria que està aprenent tot hi el sobreentrenament, però, si s'observen les mètriques *precision* i *recall* es descobreix que per culpa del baix *recall*, la xarxa neuronal experta tant en sota-entrenament com en sobreentrenament no classifica, sinó que decideix classificar totes les imatges com a modificades.

Si estudiem el millor estat de la xarxa neuronal experta (40 *epochs*) a la figura 6, descobrim que la xarxa neuronal experta ha presentat el millor resultat amb les imatges sense modificació, seguit per les imatges amb la modificació *cmb* i *lnoise*.

7.1.2 Harris Detector i Fourier Transform

Com s'ha explicat anteriorment, amb aquestes xarxes neuronals es tenia l'objectiu d'analitzar si amb una petita pèrdua d'informació i realitzant una abstracció de la imatge, la xarxa neuronal experta és capaç de tenir un comportament adequat.

Observant la gràfica 4 podem observar que la xarxa neuronal experta que tracta amb la transformada de Fourier presenta un millor resultat que la que tracta amb Harris Detector. També s'observa que les dues xarxes neuronal expertes milloren a l'augmentar l'entrenament, a diferència tant de la xarxa neuronal experta en original.

Analitzant en profunditat la gràfica de l'estat de millor rendiment de les dues xarxes neuronals 7, 100 *epochs*, observem el següent: la xarxa neuronal experta en Fourier està tenint un problema amb el sobreentrenament en la modificació de "multi paste", però no es rellevant degut al fet que hi havia pocs exemplars, igualment, descobrim que aquesta xarxa neuronal experta obté molts bons resultats en totes les modificacions restants quan sobreentrenem la xarxa neuronal, per altra banda, la xarxa neuronal experta en la característica Harris Detector, presenta molt bon resultat en totes les modificacions, però no és capaç de detectar les imatges sense modificacions amb un bon resultat.

D'aquests experiments se'n pot extraure la idea que, la pèrdua d'informació no ha estat una afectació important en la xarxa neuronal, però l'abstracció realitzada sobre la imatge sí.

7.1.3 HSV & YUV

Amb aquestes dues característiques, es plantejava l'objectiu de realitzar una transformació a la imatge que no suposes una pèrdua en la informació, per a poder estudiar el comportament de la xarxa neuronal experta respecte transformacions directes.

Fent un primer anàlisi a la figura 4, s'observa que les xarxes neuronals expertes, a mesura que s'augmenten el *epochs* presenten millors resultats, de manera similar. Però, aquest fet d'unió en comportament es contraposa directament amb l'anàlisi sobre el comportament respecte a les modificacions de les imatges en el seu millor estat, 100 *epochs* 7, en la figura 7 es pot observar que tot hi ser transformacions directes, les xarxes neuronals expertes no comparteixen resultats.

A manera de resum d'aquest experiment, se'n pot extraure que tot hi ser transformacions directes sense pèrdua d'informació de la imatge, aquestes presenten resultats diferents, de manera que s'han de considerar, ja que per exemple, si es vol detectar *multi_paste* resulta més efectiu una xarxa neuronal experta basada en YUV que en RGB o HSV.

7.1.4 Histogram

Amb aquesta característica es buscava reduir al màxim la informació d'una imatge amb l'objectiu d'estudiar-ne el comportament.
Els resultats esperats eren que, amb molt poca informació la xarxa neuronal experta no fos capaç de classificar correctament.

Sorprenentment, com es pot veure a la figura 4, aquesta xarxa neuronal experta ha donat un resultat molt bo quant s'ha augmentat l'entrenament, aquest fet refuta la idea inicial de la pèrdua d'informació, i presenta la teoria que amb suficient entrenament, una xarxa neuronal experta és capaç de donar un resultat adequat tot hi rebre un input

neural_net	epochs	ACCURACY	PRECISION	RECALL	TRUE_NEGATIVE_RATE
OriginalNN	5	0,52	0,5	0,02	0,98
OriginalNN	40	0,55	0,52	0,81	0,31
OriginalNN	100	0,56	0,83	0,1	0,98
HsvNN	5	0,53	0	0	1
HsvNN	40	0,48	0,47	0,88	0,13
HsvNN	100	0,61	0,6	0,5	0,7
HistogramNN	5	0,48	0,48	0,98	0,02
HistogramNN	40	0,59	0,64	0,33	0,83
HistogramNN	100	0,63	0,6	0,71	0,56
HarrisNN	5	0,47	0,47	0,71	0,25
HarrisNN	40	0,52	0	0	1
HarrisNN	100	0,59	0,73	0,23	0,92
FourierNN	5	0,52	0	0	1
FourierNN	40	0,54	0,54	0,31	0,75
FourierNN	100	0,78	0,84	0,67	0,88
CrominanceLuminanceNN	5	0,46	0,47	0,92	0,04
CrominanceLuminanceNN	40	0,51	0,49	0,94	0,12
CrominanceLuminanceNN	100	0,53	0,51	0,71	0,37

1 - 18 / 18

Fig. 4: Gràfica que mostra les métriques de *Accuracy*, *Precision*, *Recall* i True negative rate de cada xarxa neuronal experta subdividit en els *epochs* d'entrenament.

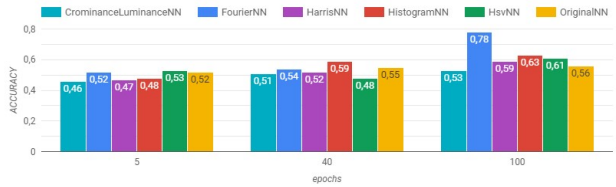

Fig. 5: Gràfica que mostra la mètrica *accuracy* de cada xarxa neuronal experta segons els *epochs* entrenats.

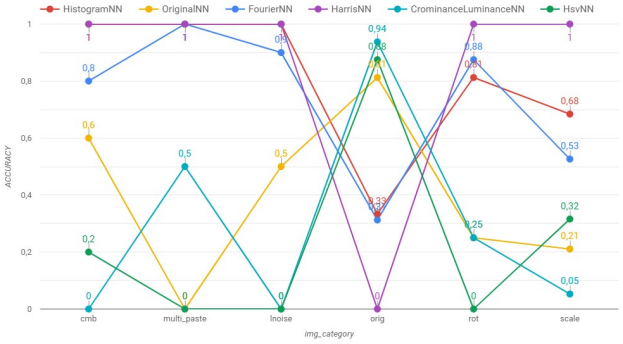

Fig. 6: Gràfica que mostra la mètrica *accuracy* de cada xarxa neuronal experta amb 40 *epochs* entrenats.

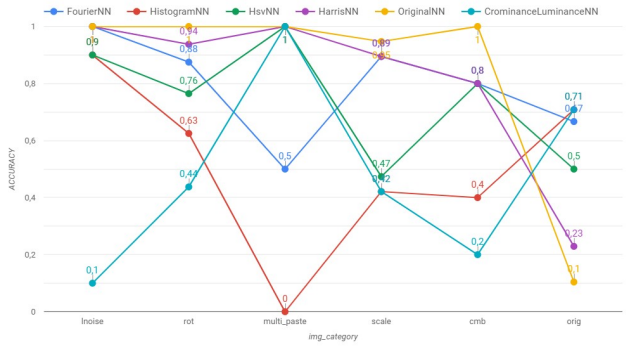

Fig. 7: Gràfica que mostra la mètrica *accuracy* de cada xarxa neuronal experta segons amb 100 *epochs* entrenats.

amb poca informació.

Realitzant un estudi sobre el comportament en vers a les modificacions, analitzant els resultats en el seu millor estat, com es pot veure a la figura 7, observem que aquesta xarxa neuronal presenta un bon resultat en la modificació lnoise però presenta un molt baix resultat en la detecció d'imatges modificades amb la tècnica de *multi_paste*.

7.2 Anàlisi composició xarxes neuronals expertes

Com es pot observar tant en la figura 9 com en la figura 8, l'estratègia d'utilitzar múltiples xarxes neuronals expertes per a donar un veredicte és més efectiu independentment de l'entrenament i de la modificació a la imatge, respecte a una sola xarxa neuronal.

Aquests experiments doncs, confirmen la teoria de la utilització de múltiples xarxes neuronals, conjuntament amb la justificació de la construcció de tota la plataforma per a la creació d'aquestes.

8 Conclusions

Com a resum i tancament del projecte, s'ha construït una plataforma que ha complert amb tots els objectius establerts i que ens permet respondre a les preguntes inicials plantejades amb el support dels resultats dels experiments realitzats en la mateixa plataforma construïda.

Respecte a l'anàlisi del comportament d'una xarxa neuronal que dóna un bon resultat en la classificació d'imatges en vers a només les característiques d'una imatge, s'ha pogut extreure la conclusió que, independentment de la pèrdua d'informació per la característica amb la qual l'entrenem, mentres aquesta xarxa neuronal experta obtingui suficient temps d'entrenament per adaptar-se a aquesta característica, donarà bons resultats.
Per altra banda, cal remarcar que no totes les característiques tot hi ser similars o equivalents en informació, com un canvi d'espectre de color, aportaran un mateix rendiment a la xarxa neuronal experta, per tant es bó realitzar un estudi sobre característiques similars en concepte.

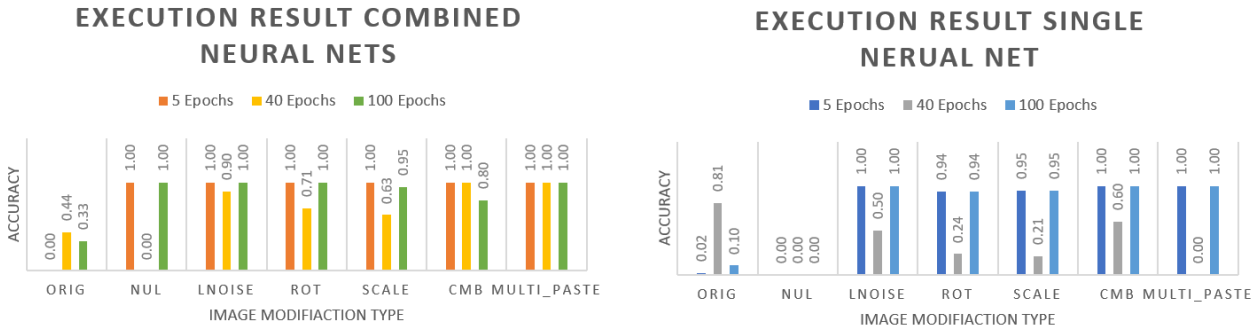

Fig. 8: Comparació dels resultats, subdividits en *epochs* i per modificació d'imatge, entre la estrategia d'utilitzar múltiples xarxes neuronals (*combined*), contra utilitzar una única xarxa neuronal (*Single*).

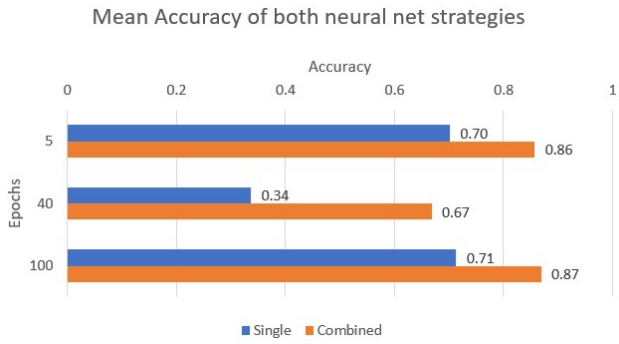

Fig. 9: Gràfica que mostra la mitjana de la mètrica *accuracy* en, *Single*, utilitzant només una xarxa neuronal experta que té com a input la imatge RGB, i *Combined*, que és la combinació de les xarxes neuronals expertes de les propietats: *Fourier Transform*, *Original*, *HSV*, *YUV*, *Harris Detector* i *Histogram*

Respecte a la hipòtesi del rendiment en vers a la detecció d'imatges modificades entre múltiples xarxes neuronals expertes i una xarxa neuronal genèrica, s'ha pogut verificar que, múltiples xarxes neuronals expertes entrenades amb característiques de les imatges aconsegueixen un millor resultat que una sola xarxa neuronal genèrica que té com a entrada la imatge en si, tot hi compartir estructura i temps d'entrenament.
Aquesta afirmació porta directament a la reconsideració de la utilització de les xarxes neuronals: potser per a millorar el rendiment de les xarxes neuronals, no s'ha de focalitzar en els paràmetres d'aquesta, sinó en quantes xarxes neuronals expertes tenim, i que n'opinen aquestes.

Agraïments

Vull tenir uns primers agraiments, a la meva familia i a la meva parella, per tot el suport, confiança i paciencia que m'han donat durant el periode de desenvolupament d'aquest projecte.
I també vull dedicar un fort agraiment al tutor Jordi Serra Ruiz per la seva dedicació i implicació en aquest projecte.

Referències

[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *CoRR*, vol. abs/1501.00092, 2015.

[2] "Divide-and-conquer algorithm," Jun 2019. Accessed: 2019-06-30.

[3] L. Breiman and E. Schapire, "Random forests," in *Machine Learning*, pp. 5–32, 2001.

[4] M. Hassaballah, A. Ali, and H. Alshazly, *Image Features Detection, Description and Matching*, vol. 630, pp. 11–45. 02 2016.

[5] Mustererkennung and F.-A.-U. Erlangen-Nürnberg, "Image manipulation dataset." Accessed: 2019-06-30.

[6] Saket, "7 best models for image classification using keras," Nov 2018. Accessed: 2019-06-30.

[7] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, July 2017.

[8] "Keras, densenet." `https://keras.io/applications/#densenet`.

[9] C. Larman and V. R. Basili, "Iterative and incremental development: A brief history," *Computer*, vol. 36, pp. 47–56, June 2003.

[10] W. Van Casteren, "The waterfall model and the agile methodologies : A comparison by project characteristics," 02 2017. Accessed: 2019-06-30.

[11] "Creating a tensorflow deep learning vm instance — deep learning vm — google cloud." Accessed: 2019-06-30.

[12] C. Harris and M. Stephens, "A combined corner and edge detector," in *In Proc. of Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[13] W. T. Cochran, J. W. Cooley, D. L. Favin, H. D. Helms, R. A. Kaenel, W. W. Lang, G. C. Maling, D. E. Nelson, C. M. Rader, and P. D. Welch, "What is the fast fourier transform?," *Proceedings of the IEEE*, vol. 55, pp. 1664–1674, Oct 1967.

APÈNDIX

Nombre de tarea	Durac	Comienzo	Fin	Predecesoras
TFG	**83 días**	**vie 8/3/19**	**lun 1/7/19**	
Estudi de les característiques	**38 días**	**vie 8/3/19**	**mar 30/4/19**	
Búsqueda de característiques	25 días	vie 8/3/19	jue 11/4/19	
Documentación de las características	10 días	vie 12/4/19	jue 25/4/19	3
Búsqueda de la extracción de la característica	10 días	vie 12/4/19	jue 25/4/19	3
Testeo correcta extracción	3 días	vie 26/4/19	mar 30/4/19	5
Construcció Dataset	15 días	vie 8/3/19	jue 28/3/19	
Construcció módul extractor de característiques	**19 días**	**vie 29/3/19**	**mié 24/4/19**	**7**
Disseny del extractor de característiques	7 días	vie 29/3/19	lun 8/4/19	
Extracció i testeig dels algoritmes d'extracció	6 días	mar 9/4/19	mar 16/4/19	9
Adaptació del dataset per a les xarxes neuronals	6 días	mié 17/4/19	mié 24/4/19	10
Xarxes neuronals expertes	**19 días**	**jue 25/4/19**	**mar 21/5/19**	**8**
Construcció d'una primera xarxa neuronal	3 días	jue 25/4/19	lun 29/4/19	
Realització del disseny modular per a les diferents xarxes neuronals	5 días	mar 30/4/19	lun 6/5/19	13
Replica de la primera xarxa neuronal per a cada característica	3 días	mar 7/5/19	jue 9/5/19	13;14
Configuració xarxes neuronals	5 días	vie 10/5/19	jue 16/5/19	15
Entrenament xarxes neuronals	3 días	vie 17/5/19	mar 21/5/19	16
Anàlisi resultats	**5 días**	**mié 22/5/19**	**mar 28/5/19**	**2;12**
Estudi efectivitat de les característiques	5 días	mié 22/5/19	mar 28/5/19	
Estudi efectivitat de les xarxes neuronals	5 días	mié 22/5/19	mar 28/5/19	
Tancament	**25 días**	**mié 29/5/19**	**lun 1/7/19**	**18**
Redactat dels resultats obtinguts	15 días	mié 29/5/19	mar 18/6/19	
Perfilat del document	5 días	mié 19/6/19	mar 25/6/19	22
Presentació oral	5 días	mié 26/6/19	lun 1/7/19	23

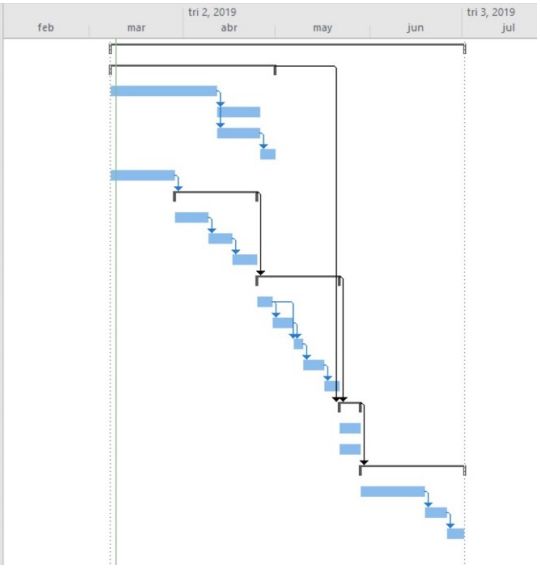

Fig. 10: Gannt referent a totes les tasques realitzades en el projecte

Fig. 11: Exemple de las diferents modificacions d'una imatge dintre del *dataset*.

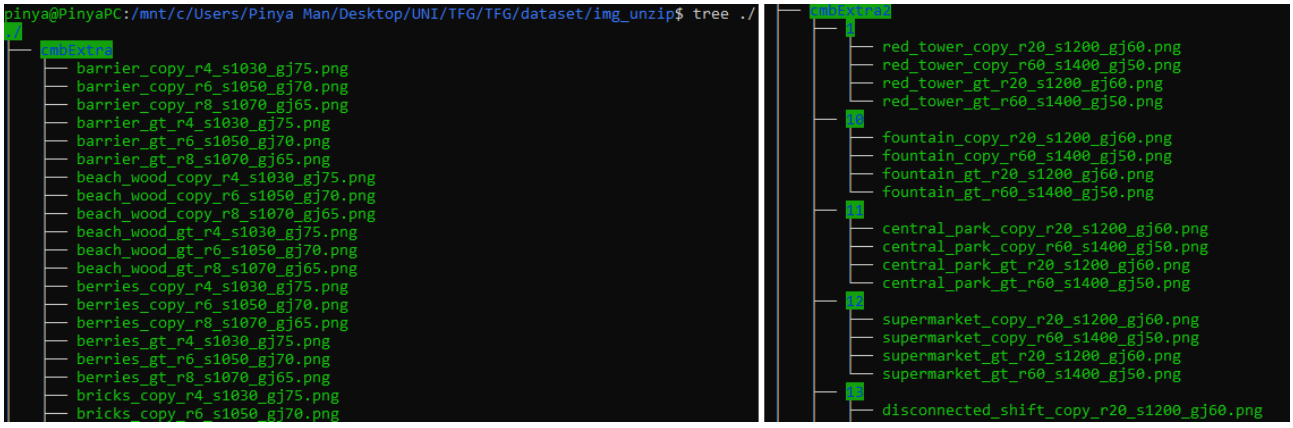

Fig. 12: Sortida per consola de la llibreria *tree* on es mostra la diferència d'estructures d'arxius del *dataset* [5], a la imatge de la dreta és abans d'aplicar el la construcció del *dataset*, i a l'esquerra després, podem observar que a la imatge de l'esquerre es compleix el conveni establert.

```
    def fourier_transform(img):
        gray_img = __get_gray_image(img)
        fft = np.fft.fft(gray_img, norm="ortho")
        return fft.astype(float)
```

Fig. 13: Exemple de funció que extreu la característica de la transformada de Fourier, d'una imatge passada per paràmetre.

```
class FourierNN(nn_abstract):
    def __init__(self, csv_filename, timestamp, save_model_output_path) -> None:
        super().__init__(csv_filename, timestamp, save_model_output_path)
        self.nn_name = "FourierNN"
        self.nn_input_size = (500, 500, 1)

    def transform_path_dataset2nn_input(self, input_path):
        image_size = (self.nn_input_size[0], self.nn_input_size[1])
        nn_input = np.zeros(
            (len(input_path),
             image_size[0],
             image_size[1]))
        for it, img_property in enumerate(input_path):
            property_np = pd.read_csv(img_property).values
            nn_input[it, :, :] = property_np[:, 1::]
        return nn_input[:, :, :, np.newaxis]
```

Fig. 14: Exemple de definició d'una xarxa neuronal experta que prediu a través de la transformada de Fourier.

Neural Net initialization parameters	
Train percent	80%
Test percent	20%
Random seed	420
Optimizer	adam
Loss	sparse_categorical_crossentropy
Metric	accuracy
Batch Size	30
Epochs	5, 40, 100

TAULA 1: CONFIGURACIÓ DE TOTES LES XARXES NEURONALS, INDEPENDENTMENT DE LA CARACTERÍSTICA.

Field name	Type	Mode
img_id	INTEGER	NULLABLE
img_path	STRING	NULLABLE
img_category	STRING	NULLABLE
real_fake	STRING	NULLABLE

Fig. 15: Esquema de la taula image_dataset.

Field name	Type	Mode
img_id	INTEGER	NULLABLE
img_path	STRING	NULLABLE
img_category	STRING	NULLABLE
real_fake	STRING	NULLABLE
property	STRING	NULLABLE
property_path	STRING	NULLABLE

Fig. 16: Esquema de la taula image_properties.

Field name	Type	Mode
img_id	INTEGER	NULLABLE
img_path	STRING	NULLABLE
img_category	STRING	NULLABLE
real_fake	STRING	NULLABLE
property	STRING	NULLABLE
property_path	STRING	NULLABLE
predicted	STRING	NULLABLE
neural_net	STRING	NULLABLE
confusion_matrix_value	STRING	NULLABLE
epochs	INTEGER	NULLABLE

Fig. 17: Esquema de la taula predictions.