Aprenent a llegir l'hora amb xarxes neuronals

Víctor Suárez Vara

Resum — Aquest article recull el desenvolupament del projecte "Aprenent a llegir l'hora amb xarxes neuronals", una recerca duta a terme amb col·laboració amb el CVC i, en concret, amb el departament de recerca en llenguatge i visió. Treballant amb xarxes neuronals dintre l'espai de treball dels rellotges, explorem la lectura d'informació d'imatges en que es requereix relacionar formes geomètriques i text (en aquest cas només números). Dintre podrem veure creacions de datasets innovadors, perfeccionament de models simples, ús d'arquitectures clàssiques preentrenades com ResNet, ús d'eines punteres per visualitzar execucions i la història d'un experiment design. Aquest projecte es en sí un doble repte, ja que pretenem aprendre a usar xarxes neuronals fent servir eines punteres començant des de una petita base de coneixement i, alhora, trobar una solució per la lectura de rellotges començant sense dades. Finalment acabem donant amb una solució parcial pel problema després de molt aprenentatge i proposant noves línies innovadores per on avançar amb aquesta investigació.

Paraules clau — rellotges, datasets, xarxa neuronal, experiment design, models.

Abstract— This article reports the development of the project "Learning to read the time with neural networks", a research carried out in collaboration with the CVC and specifically the department of research in language and vision. Working with neural networks within the workspace of clocks, we explore the reading of information from images in which it is required to relate geometric shapes and text (in this case only numbers). Inside this document, we will see innovative dataset creations, refinement of simple models, use of classical pre-trained architectures such as ResNet, use of top tools to visualize executions and the history of a design experiment. This project is a double challenge, since we intend to learn how to use neural networks using top tools starting from a very low level and, at the same time, we try to find a solution for reading clocks starting from scratch, without no data. Finally, we ended up with a partial solution to the problem after a lot of learning and proposing new innovative lines to move forward with this research.

Index Terms— clocks, datasets, neural networks, experiment design, models.

1 Introducció

1.1 Concepte general

APRENENT a llegir l'hora amb xarxes neuronals és una recerca proposada per Dimos Karatzas, tutor d'aquest treball. Aquesta tracta de llegir informació d'imatges a través de relacionar formes geomètriques i text (números en el nostre cas). Podem veure'n un exemple a la figura 1 sobre el nostres espai de treball, els rellotges.

Fig. 1: Imatge amb els elements necessaris per llegir l'hora emmarcats amb requadres vermells.

El camp dels rellotges ofereix molta variabilitat, ja que hi ha molts tipus de rellotges i usen diferents formes i estils de text, com podem veure a la figura 2, cosa que ens fa més complicat i realista el problema. A més, cal destacar que encara no existeix un programa capaç de llegir l'hora per a qualsevol rellotge.

Fig. 2: Imatge amb molts rellotges diferents.

Des del 1994, utilitzant mètodes senzills de visió per computadors ja es podia relacionar geometries i text per llegir instruments analògics com podem veure en aquest *paper* [1], i usant *improved Hough transformation* en l'última dècada ja s'havia aconseguit algorismes eficients per llegir l'hora de rellotges analògics [2], tot i això la visió tradicional està molt limitada a adaptar-se a cada espai de treball com ja comenten aquestes publicacions citades.

Amb aquest treball doncs utilitzem xarxes neuronals, que són una alternativa a l'ús de visió tradicional on ja

- E-mail de contacte: victorsuarezvara@gmail.com
- Menció realitzada: Computació
- Treball tutoritzat per: Dimosthenis Karatzas (Ciències de la Computació)
- Curs 2021

s'han aconseguit solucions parcials per a la lectura de rellotges com comentarem més endavant [3]. Durant el treball s'utilitzen tant models amb arquitectures simple, com models amb arquitectures modernes preentrenades com ResNet [4]. S'utilitzen *datasets* extrets d'internet i *datasets* creats per nosaltres per requeriments de la recerca.

Donat que llegir informació d'imatges relacionant text i formes geomètriques és un terreny amb molt per explorar, començarem enfocant el problema només a rellotges analògics, començant per dades sintètiques i avançant a dades cada cop més reals. Comencem també suposant que el número 12 sempre està a dalt i el rellotge sempre està centrat.

Destacar també, com a motivació i refutació que encara no existeix una solució per a llegir qualsevol rellotge, que els models més punters de xarxes neuronals de VQA tampoc llegeixen rellotges com es pot observar en l'article de la figura 2 [5].

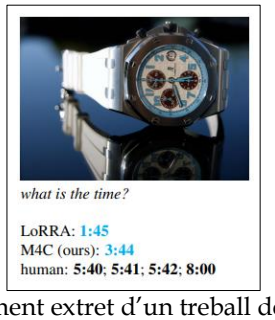


Fig. 2: Fragment extret d'un treball de *VQA* [5], que mostra que el millor model llegeix les 3:44 quant son les 5:41.

Com partim del coneixement que dóna la carrera sobre *deep learning*, que no és suficient per a saber usar aquesta tecnologia a escala puntera i funcional i, comencem sense cap dataset de rellotges, podem dir que aquest projecte és un doble-repte: aprendre a emprar xarxes neuronals i, començant sense dades, donar amb un model òptim per la lectura de rellotges.

Aquest document està organitzat per seccions, de manera que primer s'introduirà les qüestions tècniques de com hem dut a terme el projecte i després s'exposarà com ha estat el transcurs de *l'experiment design,* així com els resultats obtinguts.

1.2 Objectius del projecte

En iniciar el projecte ens vam plantejar expandir la recerca a altres instruments analògics en algun punt del projecte, però vam decidir donar prioritat al que sorgís durant *l'experiment design,* garantint-nos així, més llibertat en aquesta fase. Els objectius del projecte doncs, són:

1. **Aprendre més sobre xarxes neuronals.**
   Es busca anar un pas més enllà i aprofundir més en els detalls intrínsecs que creen les xarxes neuronals així com conèixer el que s'usa en la indústria al nivell més punter possible.
2. **Aprendre a resoldre problemes reals amb xarxes Neuronals.**
   Saber utilitzar xarxes neuronals simples, arquitectures prentrenades, saber com modificar els hiperparàmetres mínimament, saber utilitzar eines per visualitzar-ne els resultats...
3. **Crear datasets**
   Aprendre a crear *datasets*, tant sintètics com realistes. Durant l'*experiment design* en construirem varis.
4. **Fer servir una xarxa neuronal per solucionar el problema**
   A partir d'un *dataset* simple de rellotges analògics, busquem trobar o crear com a mínim un model que en presenti overfitting per demostrar-nos que sabem usar aquesta tecnologia per solucionar problemes reals.

2 Estat de l'art

Pel que fa al disseny de l'arquitectura del *baseline model* que s'usa com a base sobre la qual iniciar l'*experiment design*: està inspirat fonamentalment en un projecte elaborat per *Shiva Verma. Reading Clocks using Neural Nets* [3] que utilitza una arquitectura senzilla per a resoldre un *dataset* de rellotges analògics sintètics. Existeixen treballs similars a l'esmentat oberts a internet [6] però, cap d'ells va un pas més enllà de rellotges analògics sintètics. Els treballs que utilitzen visió tradicional [1], [2], com usem xarxes neuronals; els deixem de banda tot i que ens serveixen d'ajuda per poder intuir, o inclús saber, que podem necessitar Quant a arquitectura. L'arquitectura que usem pel *baseline,* consta de quatre capes convolucionals seguides d'una divisió del flux en dues branques, una per classificar hores i l'altre pels minuts, amb dues capes *fully connected* cadascuna.

Per aquest treball es clau l'ús de *datasets* de qualitat i donat que a internet existeixen diferents *datasets* de rellotges, tot i que tots ells sintètics; i no hi ha cap d'ells que sigui diferencialment millor que la resta, s'utilitza el dataset que utilitza *Shiva Verma* pel *baseline* [3], així facilitem arribar a assolir una base sòlida sense gaire dificultat. Durant les fases posteriors del treball, dintre *l'experiment design,* es creen els *datasets* per adaptar-los a la recerca que es dur a terme i per complir l'objectiu 3 de crear *datasets.* Quan arribem al punt durant el projecte en que necessitem accés a més *data,* o se'ns quedin curts els nostres *datasets*, utilitzarem arquitectures preentrenades clàssiques com *ResNet* [4] per reaprofitar les features amb les quals s'ha entrenat de *ImageNet*.

En cada pas que anem fent dintre l'*experiment design*, s'analitzarà què existeix al mercat o que s'ha realitzat per evitar reinventar la roda com veurem a l'apartat de metodologia. Així, per exemple, en la tria de valors pels hiperparàmetres, s'agilitza la cerca dels millors valors usant valors que es recomanen en publicacions, com el valor de la regularització L2 triat de 0.0005, presentat com el millor valor en aquest *paper* [7].

També utilitzarem tècniques com learning rate decay o l'ús de *LeakyReLu* com a funció d'activació, extretes de la realització dels cursos de l'especialització de *Coursera* [8].

Per últim, també s'usen lectures de publicacions i notícies per trobar la inspiració per determinar per on avançar durant el transcurs del projecte, obrint la ment a noves idees i línies de treball. Com amb la paradoxa de *Moravec,* "la intel·ligència artificial fa fàcil el que és difícil (i viceversa)" [9] o la lectura del llibre de *Michael A. Nielsen* que demostra que una xarxa neuronal es teòricament capaç de solucionar qualsevol problema si es pot expressar com una funció [10].

3 Viabilitat del Projecte

3.1 Requisits hardware

Per a dur a terme aquest projecte tan sols es necessita accés a entorns d'altes capacitats de còmput (GPU). S'usa l'entorn gratuït *Col·lab* de Google Drive, un entorn que permet córrer *notebooks* en línia a GPUs de *Google* i un ordinador del CVC amb una GPU 1080.

3.2 Requisits software

Tots els recursos que es requereixen software per aquest treball són gratuïts. Es necessita un entorn on programar, un llenguatge de programació i llibreries de qualitat. En concret s'usa la llibreria de *PyTorch* de *python*, que és una de les més utilitzades pels investigadors i la que s'usa al CVC.

3.3 Requisits d'aprenentatge

Per aprendre a usar xarxes neuronals es necessita accedir a informació de qualitat, hi ha molt a prendre i poc temps. La especialització de *Coursera* en *deep learning* [8] es la més adient per aprendre correctament les bases en detall i, com que l'aprenentatge de més qualitat és aquell impartit per umans, es requereix de l'ajuda de gent experimentada en el sector, en aquest cas estudiants i investigadors del CVC i el tutor del treball Dimosthenis Karatzas.

4 Planificació

El projecte consta de cinc fases principals resumides a continuació:

1. **Aprendre en profunditat que son les xarxes neuronals.**
   Aquesta pretén dotar a la investigació d'una base ferma de coneixement complint el primer objectiu del treball. Entre altres publicacions i llibres [9],[10], es realitzaran *Courseras* del programa de *deep learning* [8]. Aquesta fase suposa unes **50 hores** tot i que s'estendrà durant tot el projecte sobretot per l'*experiment design*.

2. **Desenvolupar un baseline model**
   L'objectiu d'aquesta fase és crear el primer model que resol el problema amb un accuracy acceptable sobre un *dataset* sintètic senzill. Serà la base a partir de la qual avançar a prototips més complexos. Aquesta fase suposa unes **40 hores**.

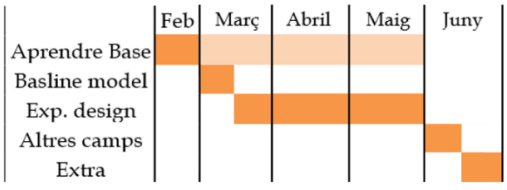
3. **Experiment design**
   L'*experiment design* se sap com es comença per no on et portarà. Dintre aquesta fase s'usa un mètode àgil amb reunions amb el tutor setmanal per compartir resultats i debatre els nous passos. Es creen *datasets* sintètics i reals, es proven diverses arquitectures, es perfecciona el model, es treballa amb eines de visualització com *Wandb*... Aquesta fase suposa unes **200 hores**.

4. **Expandir a altres camps (fase polivalent)**
   Aquesta fase no la fixem com a obligatòria per dotarnos de certa llibertat en cas que durant l'*experiment design* ens sorgeixi alguna nova línia de treball. En aquesta fase es pretén fer servir el model més realista creat a la fase anterior per resoldre un problema semblant al que ataca aquest projecte i suposa unes **40 hores**.

5. **Extres (fase polivalent)**
   Aquesta fase també es polivalent i està destinada a l'exploració d'idees que hagin sorgit durant el transcurs del treball si sobressin hores.

La planificació temporal d'aquestes fases es pot representar gràficament en un *Diagrama de Gantt*, com el que es pot veure a la Taula 1.

TAULA 1: Diagrama de Gantt simplificat



5 Metodologia

La metodologia usada durant aquest projecte la dividim en la metodologia usada durant *l'experiment design* i la metodologia emprada durant la resta de fases.

Durant les fases d'aprenentatge i construcció del *baseline*, hem realitzat una previsió total de les hores que suposaria i hem planificat al detall l'execució de les tasques. Un cop per setmana es duia a terme una reunió amb el tutor per avaluar el progrés.

Per l'experiment design, a causa de la seva naturalesa iterativa, durant el transcurs del projecte hem fet reunions setmanals amb el tutor, per poder veure la progressió, i definir nous objectius setmana a setmana. Per adaptar-se fàcilment als canvis hem seguit una metodologia *Àgil* [11]. Donat que les xarxes neuronals són un camp on hi ha molta desconeixença, augmentar la complexitat lentament és vital per ajudar a la comprensió. Per exemple si es complica el *dataset* afegint-li dues millores a les imatges, com perspectiva i soroll, si s'obtenen mals resultats no es pot saber si és perquè el model no sap lidiar amb la perspectiva o si el model no sap lidiar amb el soroll. Augmentar la complexitat lentament ha estat vital doncs.

Així doncs, la metodologia d'aquesta fase consisteix en un procés iteratiu com el que es mostra a la figura 4, que es suporta sobre les reunions setmanals amb el tutor i que dura fins al final del projecte augmentant la complexitat lentament.

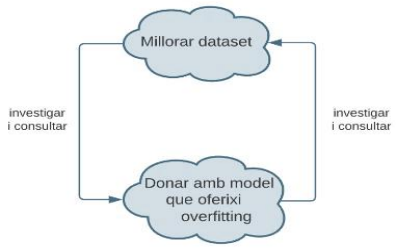


Fig. 4: Procés iteratiu usat per l'*experiment design*.

6 Riscos del Projecte

Aquest projecte l'hem plantejat de forma que els riscos siguin mínims. Per això en les primeres setmanes, ja es va planejar assolir tots els objectius plantejats a través del *baseline model*. El risc principal era no assolir aquest *baseline* que estava planejat per la fase 2 i en el pitjor dels casos, simplement, suposaria més temps que extrauríem de les últimes fases polivalents. A més a més, teníem la certesa que es podia dur a terme donat que existeix un projecte similar [3]. Aquesta minimització dels riscos ens ha donat llibertat per a realitzar un *experiment design* molt lliure i anar decidint a mesura que avançàvem.

7 Experiment Design

Després de 3 cursos de Coursera sobre *deep learning* [8], aprendre a fer servir *Col·lab*, *PyCharm* i *PyTorch*, aprendre a fer servir GPUs i moltes lectures de papers, articles i llibres electrònics sobre xarxes neuronals, comencem amb l'experiment design, la fase on es desenvolupa la major part del treball.

Aquest apartat està dividit en seccions ordenades temporalment segons els experiments i dintre de cada secció es seguirà una estructura similar dividint el contingut en *dataset* emprat, model i resultats i conclusions.

7.1 Baseline

Abans de començar l'*experiment design*, el pas 0 és assolir una base sobre la qual començar, un *baseline* model. Aquest no ha estat fàcil d'assolir, tot i que parteix de replicar el treball realitzat per *Shiva Verma* [3] adaptat a *PyTorch*. La complexitat per aconseguir fer servir una xarxes neuronal correctament sense haver treballat amb la llibreria *PyTorch* abans ha estat alta.

7.1.1 Dataset emprat

Dataset sintètic extret de *Kaggle* que conté 50000 imatges on es donen totes les combinacions d'hores i minuts amb igualtat. Més informació a la web de *Kaggle* [12]. Se'n pot veure un mostreig a la figura 5.

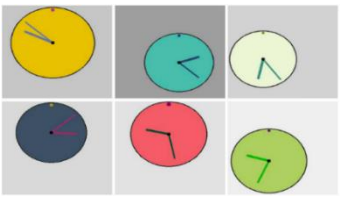


Fig. 5: *Dataset* sintètic de imatges de rellotges.

És el dataset més senzill utilitzat durant el projecte, i l'única peculiaritat que té és que els rellotges estan a escales diferents i son de diferents colors. Tots els rellotges tenen el "12" a la mateixa posició i no tenen perspectives.

7.1.2 Model

L'arquitectura és conformada per una part de visió amb 4 capes convolucionals i dues capes *fully connected* per la categorització d'hores i minuts. Podem veure un esbós general a la figura 6.

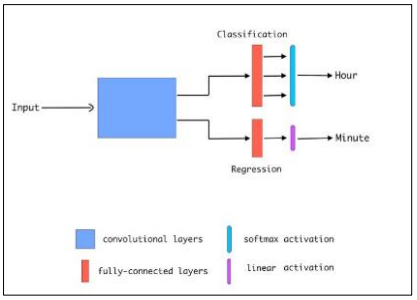


Fig. 6: Arquitectura del *baseline* model general.

Més en concret, fem servir sèries d'una capa convolucional, seguida d'una activació *ReLu*, seguida d'un *MaxPool* i aplicant-hi *batch normalization* abans de la pròxima sèrie. A la 4ª i última sèrie convolucional fem una convolució seguida d'un *batch normalization* i un *dropout* per accelerar la conversió. Se'n pot veure un esbós a la figura 7.

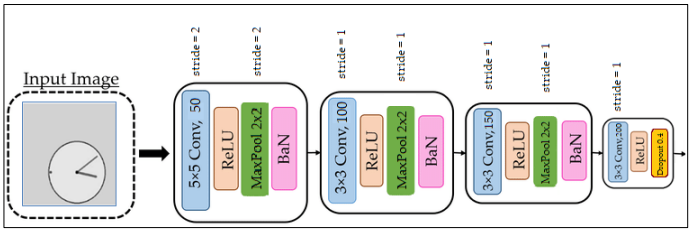


Fig. 7: Capes convolucionals del *baseline* model al detall.

Acabades les capes convolucionals es divideix el flux en dues branques per categoritzar adequadament les hores i els minuts. Per les hores usem classificació, donat que només hi ha 12 possibilitats; i. pels minuts, usem regressió, ja que hi ha 60 possibilitats i usant classificació tardaria més a convergir, encara que poc. Per classificar les hores com a mètrica usem accuracy i pels minuts fem servir una mètrica anomenada MAE (*mean average error*), que ofereix la mitjana d'errors sobre les prediccions. Mostrem un historiograma sobre aquesta mètrica per facilitar la comprensió.

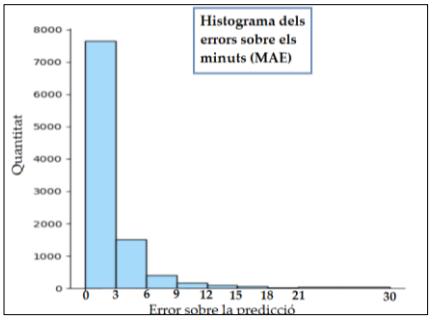


Fig. 8: Historiograma de l'error sobre els minuts

TAULA 2: RESULTATS DE LES EXECUCIONS

Model - dataset	Accuracy hores			Mae (mean average error) minuts		
	Train set	Test set	Test real set	Train set error	Test set error	Test real set error
Basline - sintètic	*99.18%*	*96.10%*	-	*3,864*	*4,356*	-
Basline – més realista	*96.3%*	*94.5%*	-	*3,96*	*4,26*	-
Well basline – diferents rellotges	*100%*	*95 %*	*10%*	*1,44*	*2,46*	*16,8*
Well basline – textures	*83%*	*13 %*	*10%*	*1,8*	*12,6*	*18,72*
Res Net – textures	*100%*	*61 %*	*16%*	*0,72*	*4,74*	*16,68*
Well basline –rellotge UAB	*100%*	*98 %*	*4%*	*1,68*	*2,1*	*16,2*

Per la classificació de les hores s'usen dues capes de 144 i 144 *features* respectivament i per la regressió dels minuts s'usen dues capes de 100 *features* cadascuna. S'usen poques capes, ja que s'espera només que relacioni la informació que prové de les capes convolucionals amb les sortides.

Més detalls de l'arquitectura:
- Input 100 x 100 x 1 normalitzat (normalització calculada fent el mean i el std de tot el training set)
- Optimització: SGD amb momentum=0.9 i *lr*=0.01
- *Training* size= 40.000 i *test size* = 10.000 imatges
- Batch size = 256
- Epochs = 20

Per més detalls accedir al notebook que conté aquest model a *Col·lab* a través del següent enllaç: https://colab.research.google.com/drive/1pMY_5KVzplGUeonhD6MPSsIOWY4lvFi3?usp=sharing

7.1.3 Resultats i conclusions

Hem aconseguit bons resultats com es pot veure a la taula 2, amb un accuracy del 96% per les hores i un MAE de 3,9 errors de mitjà pels minuts sobre el test set. Podem veure l'ouput directe sobre una imatge a la figura 10. També hem provat d'usar arquitectures prentrenades com VGA o AlexNet i els resultats han estat molt pitjors que amb la arquitectura simple presentada.

Tot i no haver pogut replicar per complert el model de *Shiva Verma*, hem donat amb un model que sap llegir l'hora d'un rellotge sintètic. Aquest model serà el nostre *baseline* model. Pròxim pas: provar el model per un *dataset* més real per veure si és capaç de llegir l'hora usant *features* reals.

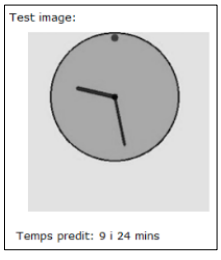


Fig. 10: Predicció del *baseline* model. El model prediu que son les 9 i 24 mins quan son les 9 i 28 mins.

7.2 Primer experiment. Dataset més realista

Aquest experiment pretén descobrir si l'arquitectura segueix sent útil a un espai de *features* reals. El primer pas doncs, és trobar o crear un *dataset* més realista.

Per internet no vam trobar-ne cap, tot i mesos més tard vam trobar un projecte que va crear un *dataset* fent fotos a un rellotge durant 12 hores [13] i el podríem haver reutilitzat i estalviat cert temps, va ser un error de estat de l'art greu; també vam descartar fer fotografies a un rellotge durant 12 hores perquè comportaria molt temps... Finalment, davant la necessitat, vam idear el següent:

7.2.1 Dataset emprat

Dataset creat extraient frames d'un vídeo d'internet d'un rellotge funcionant durant 12 hores en format *time-lapse* (durava 1min). L'etiquetació d'hora i minuts és automàtica donat que les mantetes avancen a la mateixa velocitat sempre i podem aplicar una màscara per obviar el fons perquè el rellotge està estàtic (màscara creada amb el paint sobre un *frame* qualsevol). S'usen *backgrounds* que s'assemblin a casos real on es podria trobar un rellotge, en concret s'utilitza un *dataset* extret d'un paper de reconeixement d'escenes d'interior, imatges d'oficines, dormitoris, cuines i classes [14]. El resultat son 50000 imatges molt reals on es donen totes les combinacions entre hores i minuts amb igualtat, com es mostra a la figura 11.

Fig. 11: *Dataset* sintètic més real d'imatges de rellotges extret d'un *time-lapse*.

El pseudocodi iteratiu del generador de rellotges es:
1. Llegir un *frame*
2. Aplicar la mascara per obviar el fons
3. Aplicar les transformacions desitjades (en aquest cas només augmentar la mida dels rellotges)
4. Seleccionar un *background* aleatori
5. Fer un *paste*

Per generar aquest *dataset* ha calgut aprendre a treballar amb vídeos, frames, CSVs, paint, llibreries d'imatge... Hem invertit al voltant de 15 hores per crear un programa que genera rellotges automàticament.

7.2.2 Model

Usem el *baseline model* amb unes petites variacions per adaptar-nos al nou espai:
- Input 256 x 256 x 1 normalitzat (normalització calculada
- Epochs = 40 (tarda massa en convergir...)
- Lr manual decay

7.2.3 Resultats i conclusions

Hem aconseguit bons resultats com es pot veure a la taula 2, però el model tarda molt més que el *baseline* model en convergir, tot i que hem realitzat diverses variacions de l'arquitectura per aconseguir una conversió més ràpida realitzant proves fallides sobre: ús de diferents funcions d'activació, ús de menys capes convolucionals, ús de més capes *fully Connected* i ús de diferents funcions d'optimització. Inspirats en l'estudi realitzat dels cursos del primer curs de *Coursera* [6] s'usa *lr manual decay*, ja que millora la conversió considerablement. També hem preentrenat el model amb la execució del model sobre el dataset sintètic però els resultats han estat pèssims tant congelant les capes convolucionals com congelant la part *fully*...

Tot i això, hem donat amb un model que sap interpretar features reals i sap llegir l'hora d'un rellotge real. Hem descobert que només calen 4 capes convolucionals, ja que les imatges no tenen perspectives i que un ordinador és molt més eficient que un humà en llegir l'hora d'un rellotge. L'ordinador només necessita localitzar a través d'uns pocs píxels la direcció de les manetes.

Aquesta diferència entre el *modus operandi* humà i de l'ordinador ens genera un dubte que llencem a l'aire:

Sabent que el cervell humà té una capacitat superior a qualsevol ordinador, si existeix un mètode més senzill per llegir l'hora, per què els humans no l'estem fent servir? És que som ineficients en algunes tasques o al contrari? Potser, per algun motiu, realment la forma més eficient a la llarga és la que fem els humans, en aquest cas localitzar les manetes... Sent això cert, aleshores, la millor innovació hauria d'anar en la línia de replicar maneres de fer dels humans tot i que trobem maneres aparentment millors.

Tornant a l'experiment, aquest ens ha obert la ment a noves possibilitats i ens han sorgit dues línies de treball que conformaran l'experiment 2 i 3. Pròxim pas: provar si el model es capaç d'aprendre de dades amb més d'un rellotge diferent.

7.3 Segon experiment. Dataset amb diferents rellotges

7.3.1 Dataset emprat

Dataset amb 5 rellotges reals diferents extrets de *time-lapses*. Creat reutilitzant bona part del codi de l'anterior *dataset*. Conté 50000 imatges on es donen totes les combinacions entre hores i minuts amb igualtat com es mostra a la figura 12.

Fig. 12: *dataset* sintètic amb diferents tipus de rellotge real

Paral·lelament veiem necessari tenir un petit conjunt d'imatges de rellotges 100% reals per veure com actua el nostre model davant casos reals, així que es crea el *dataset* real. Aquest l'utilitzem en totes les execucions dels models a partir d'ara com a conjunt sobre el qual fer test real i saber si el nostre model evoluciona cap a casos reals o no. Les fotografies han estat extretes d'internet i han estat anotades i retallades a mà. Com es pot veure a la figura 13, les imatges són fotografies frontals sense perspectives i a diferents escales. És un *dataset* petit de 87 imatges on no es donen totes les combinacions entre hores i minuts amb igualtat.

Fig. 13: Petit *dataset* real d'imatges de rellotges anotat a mà per fer test real sobre els models.

7.3.2 Model

En aquest experiment es dona un gran salt qualitatiu. Després de realitzar més de 200 proves es millora el *baseline model* passant de tardar 50 minuts a convergir, a tardar 20 minuts. A l'anterior experiment va quedar clar que el *baseline model* tenia mancances i per fer front a aquest nou experiment era necessari un salt evolutiu. Aquest nou model l'anomenem *well baseline*.

Well *baseline*

Hem netejat i optimitzat el codi considerablement: abans transformàvem les imatges en cada lectura, reduint-les de mida i transformant-les a escala de grisos utilitzant molt còmput innecessari. Ara guardem les dades a la corresponent carpeta amb el format que utilitza la xarxa neuronal (256x256x1). Gràcies a l'ús de una eina *Wandb* per visualitzar las execucions de xarxes neuronals, hem descobert que el problema de la lenta convergència residia en un error de programació al sumar la *loss* de la branca de regressió i la *loss* de la branca de classificació, com que usen diferents mètriques estaven en rangs diferents. També usem les activacions sempre amb l'*inplace* activat i en comptes de *ReLu*, utilitzem *LeakyReLu* que ajuda una mica a convergir. S'utilitza un *batchsize* de 512 optimitzant l'ús de la GPU.

Pel que fa a millores, inspirats en un dels cursos de l'especialització de Coursera, per millorar el temps que tarda en convergir, usem *learning rate decay* automàtic. Aquest s'activa a partir de la *epoch* 15 i quant detecta que s'està estancant la *loss*, redueix el *learning rate*.

Altrament, inspirats en un *paper* que estudia diferents valors de regularització sobre un altre camp que no es d'imatges, recomana l'ús d'un valor per a la regularització molt mínima de 0.0005 que es el que hem provat i ha ofert millors resultats que sense regularització[6].

Llegint a fons les diferents opcions de funcions d'acti-

vació que oferia PyTorch a la seva web vam trobar l'activació *HardSigmoid*, aquesta funció col·loca els valors entre 0 i 1 donant les mateixes possibilitats a tots els valors, com podem veure a la figura 14.

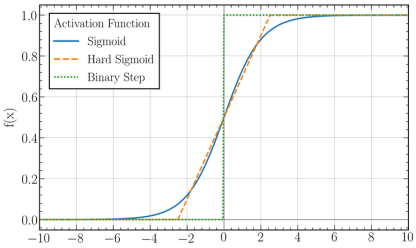
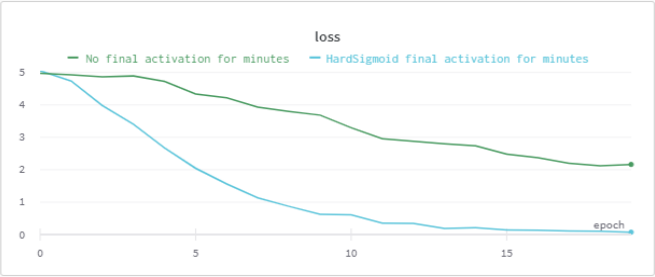


Fig. 14: Activació *HardSigmoid.*

Dintre la part *fully* dels minuts (la regressió), hem descobert que utilitzant aquesta activació per la última capa el model convergeix millor. Això es degut a que el *criterion* que s'usa per la regressió espera dades entre 0 i 1 que es el que, precisament, fa aquesta activació.

Podem veure a la figura 15 una mostra de les dos execucions. Aquest mètode de comparar execucions de la *loss* es el mètoque s'ha usat per incorporar millores al model.

Fig. 15: *Loss* amb l'ús de *HardSigmoid* i sense.



Gràcies a aquest avenç, i la part *fully Connected* es redueix, assa de 2 *layers* per branca a tant sols una layer per les hores de 128 *features* i una *layer* pels minuts de 64 *features*. La part convolucional la mantenim intacta.

Hem realitzat també diverses proves fallides provant d'utilitzar regressions per les dues branques i classificació per les dues branques. Ambdues les hem desestimat en el cas de les classificacions perquè comporten més temps per convergir i en el cas de les regressions perquè fa més difícil la conversió de les hores ja que els valors estan massa pròxims. Finalment el més eficient es usar una regressió pels minuts i una classificació per les hores. També hem provat l'ús de *Nestorov*, un accelerador del optimitzador SGD. Aquest com que arrisca més, a vegades impulsa l'algorisme a saltar mínims locals i a vegades crea execucions més lentes ja que s'equivoca. Destinem l'ús de *Nestorov* a priori perquè no es necessari per a aquest espai de treball.

Hem descobert també que aquesta xarxa té més capacitat de la que necessita perquè hem provat de reduir la mida dels *kernels* de totes les capes considerablement i, amb moltes més *epochs (60),* ha aconseguit fer *overfitting*.

Hem donat dons amb un model ràpid i òptim per a llegir l'hora dels *datasets* que tenim a l'abast. Per saber més detalls dels hiperparàmetres escollits consulteu el *GitHub* del projecte, compartit a l'apartat de resultats.

7.3.3 Resultat i conclusions

El model ha après de les dades correctament aconseguint bons resultats tot i que pel conjunt de test real els resultats són pèssims com podem veure a la taula 2. Cal destacar que hem obtingut millors resultats dels quals proposava l'article de Shiva Verma [3] sobre el qual es va crear el *Baseline model*.

El model sap llegir rellotges reals diferents només si ha après d'ells. Hem descobert que les xarxes neuronals aprenen de les features de les dades amb les que s'entrenen i si a una xarxa neuronal li donés un cas amb *features* que no ha vist mai, no sabrà resoldre el problema. Aquí doncs ens apareix un gran poder i limitació de les xarxes neuronals. Pròxim pas: forçar el model a aprendre a llegir sense usar les *features*.

7.4 Tercer experiment. Dataset de textures.

Aquest experiment pretén donar a la xarxa neuronal un *dataset* de rellotges molt diferents per forçar la xarxa a que trobi una manera de solucionar el problema sense usar *features.*

7.4.1 Dataset emprat

Dataset de rellotges sintètics creats amb un programa fet per nosaltres, en que cada rellotge té textures sempre diferents per las manetes i la circumferència.

Per fer més semblant a la realitat l'espai hi ha rellotges amb maneta de segons i rellotges sense manetes de segons, així com diferents tipus de manetes de l'hora i s'utilitza un *dataset* de textures exageradament molt variades. Aquest *dataset* conté rellotges més complicats de llegir que els rellotges reals, com podem veure a la figura 16. Esperem que si una xarxa pot resoldre rellotges més difícils que els reals pugi resoldre rellotges més senzills un cop s'hagi arribat al *overfitting*.

Fig. 16: *Dataset* sintètic de imatges de rellotges molt diferents creat amb textures.

Conté 50000 imatges on es donen totes les combinacions entre hores i minuts amb igualtat, es genera automàticament i el pseudocodi iteratiu que genera aquest *dataset* es:

1. Seleccionar una hora i minut
2. Seleccionar una textura aleatòria per les manetes d'hores, minuts i segons
3. Seleccionar una textura aleatòria per la circumferència
4. Seleccionar un background aleatori
5. Fer els corresponents pastes

7.4.2 Model

Per resoldre aquest *dataset* hem utilitzat el *well baseline model,* i una ResNet amb preentrenament [4]. Com es sabut, la ResNet està entrenada amb les imatges de *ImageNet* i doncs ja conté algunes de les features que demandem, cosa que esperem que faci més senzill pel programa arribar a ov*erfitting.* Podem veure l'arquitectura empleada a la figura 17. S'usa una ResNet50 amb preentrenament per la part convolucional. Aquesta te congelades les primeres 4 capes i la última descongelada. La part fully connected es molt similar a la utilitzada al *well baseline model* i té només una layer amb 256 *features* per la classificació de les hores i 64 *features* per la regressió dels minuts amb *LeakyReLU* i *HardSigmoid* a l'última capa de la regressió. Per veure més detalls consulteu el *GitHub* compartit a l'apartat de resultats. Destacar que hem usat Adam.

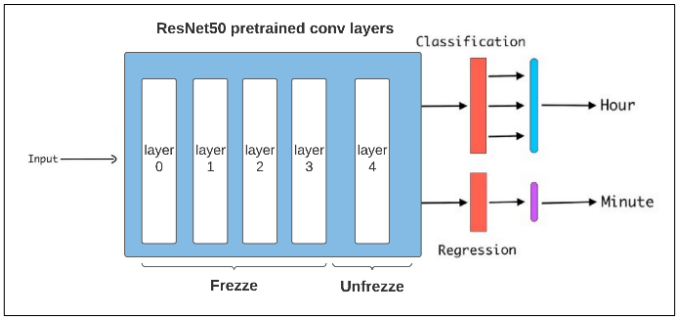


Fig. 17: Arquitectura del model utilitzant ResNet empleat per llegir l'hora d'imatges de rellotges.

7.4.3 Conclusions i resultats

En línies generals, podem dir que l'experiment ha estat fallit. Esperàvem que la dificultat afegida al *dataset,* que cada rellotge fos diferent l'un de l'altre, portes la xarxa a aprendre de forma diferent però no ha estat així. Contràriament a aquesta intenció, hem vist com el *well baseline model* no te suficient capacitat per arribar a fer *overfitting*, com es pot comprovar a la taula 2 i a la figura 18.

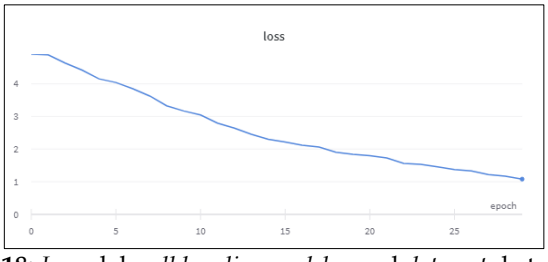


Fig. 18: *Loss* del *well baseline model* per el *datasest* de textures, que baixa molt lentament durant 30 *epochs* sense arribar a fer *overfitting*.

I hem acabat utilitzant una ResNet amb més capacitat i amb preentrenament per solucionar el problema aprenent de les features. Hem arribat a fer *overfitting,* com podem veure a la taula 2. Tot i això, la diferència entre els resultats del *traint set* i el *test set* en els dos models, ens indica com clarament no s'està entrenant correctament per aprendre sense usar les features i això comporta que els resultats en ambdós conjunts de *test* siguin pitjors. Algunes de les features dels rellotges del *test set* no les ha vist mai abans.

Durant aquest experiment ens ha sorgit la intriga de saber que deu estar succeint dintre les capes convolucionals i inspirats en *papers* com aquest [15], i hem realitzat una funcionalitat per fer *plots* de qualsevol de les capes de la xarxa neuronal. Amb aquesta funcionalitat el hem pogut veure com realment es incomprensible a simple vista que significa la informació que passa d'una capa a un altre. Petits experiments com aquest ens ajuden a entendre i refermar-nos en la idea que les xarxes neuronals es fonamenten en el concepte de lliure albir i que més que pensar en evolucions de models que passen per limitar la xarxa, es més adient pensar en millores que parteixen d'aquest grau de llibertat.

No hem aconseguit fer que la xarxa neuronal aprengui d'una altra manera, però ara sabem el potencial que tenen les xarxes neuronals, aprendre de les *features* amb les que les entrenes. Altrament seguim sabent que existeix una manera de no aprendre de les *features*, inspirats en que les xarxes neuronals, en teoria, com remarca el llibre de *Michael A. Nielsen*, "Poden resoldre qualsevol funció" o cosa equivalent, qualsevol problema [10].

Arribats aquest punt, anàvem enrederits de la planificació pel fet que l'últim experiment ens va portar més temps del que havíem planejat, vam haver de triar si continuar amb la planificació prevista i executar les fases polivalents expandint el problema al cas dels indicadors de pressió de fàbriques o prendre un altre camí. Finalment, vam decidir continuar amb la investigació de la lectura de rellotges fent servir xarxes neuronals, desviant-nos de la planificació per a concloure el treball amb aquest últim experiment. Pròxim i últim pas: solucionar el problema per a qualsevol rellotge.

7.5 Quart i últim experiment. Rellotge de la UAB.

Hem aprés que una de les potències de les xarxes neuronals es que aprèn de les features amb les que s'entrena, així que aquest experiment pretén resoldre qualsevol rellotge a partir de crear un *dataset* del rellotge i després entrenar un model amb aquestes dades per saber llegir aquell rellotge.

Com en l'experiment anterior vàrem crear un generador de data que a partir d'una imatge d'una circumferència i una imatge de cada maneta creava infinits rellotges analògics automàticament anotats, ara, usem aquest mateix algorisme per a partir d'una imatge d'un rellotge crear un *dataset.* Tant sols hem de separar les manetes de la circumferència de la imatge i després adaptar-les al programa generador amb Paint mateix.

Hem triat, per a crear l'exemple, un rellotge de les classes d'Enginyeria de la UAB per donar-li més de proximitat al projecte.

7.5.1 Dataset emprat

A partir d'una imatge d'un rellotge de les classes de la UAB, hem extret les manetes de la circumferència usant eines d'edició professionals (*Paint* i *Word)* i hem creat un programa generador reutilitzant bona part del codi de l'anterior generador del *dataset* de textures. Finalment hem creat un *dataset* de 50000 imatges on es donen totes les combinacions entre hores i minuts amb igualtat, podem veure'l a la

figura 19.

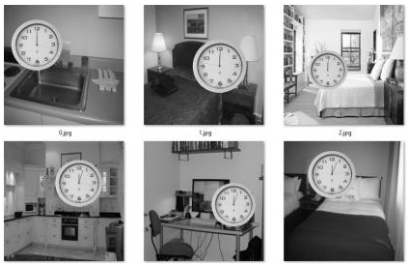


Fig. 19: *Dataset* de rellotges de la UAB, amb blanc i negre que es com entren les imatges a la xarxa neuronal.

7.5.2 Conclusions i resultats

El model ha après de les dades correctament aconseguint bons resultats com podem veure a través de la *loss* de la figura 20, tot i que pel conjunt de test real els resultats també són pèssims com podem veure a la taula 2.

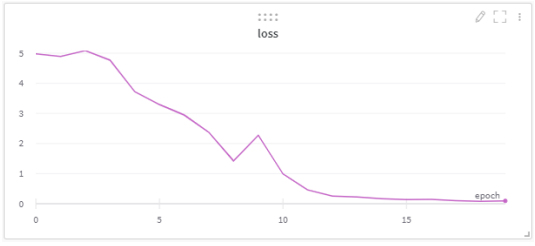


Fig. 20: Loss del well *baseline* model sobre el *dataset* del rellotge de la UAB visualitat amb l'eina *Wandb*.

Aquest experiment ens demostra que podem fer el mateix amb qualsevol rellotge i entrenar després una xarxa neuronal perquè pugui llegir-lo. Això ens obre la porta a la idea de poder crear un dataset massiu i ens demostra que el *well basline model* es funcional per a la lectura de rellotges.

Amb aquest experiment finalitzem l'*experiment design,* la fase durant la qual hem après més sobre el profund món del deep learning. Han sigut molts els experiments realitzats i només hem seleccionat els millors o més interessants, i tot i això, han quedat molts experiments per fer essencials com tocar perspectives o afegir soroll.

8 Resultats

Els resultats d'aquest projecte es poden mesurar en contribucions realitzades al camp de la lectura de rellotges.

Pel que fa a *datasets*, durant aquest projecte hem realitzat 3 generadors de *datasets* diferents que estan oberts *open source* a través de *GitHub* i hem realitzat 4 *datasets* útils per a l'entrenament de xarxes neuronals per a la lectura de rellotges que estan pujats a *Kaggle,* al següent enllaç:

https://www.kaggle.com/vctorsuarezvara/datasets

Respecte a les xarxes neuronals, gràcies a realitzar més de 50 programes de xarxes neuronals, a crear un model des de zero (sense utilitzar llibreries), a realitzar més de 500 execucions de xarxes neuronals entre *PyCharm* i *Col·lab* i tocar diferents arquitectures; hem obtingut dos models funcionals per a la lectura de rellotges: el *well baseline model* i el que utilitza *ResNet* amb preentrenament. Ambdós s'han obert al públic *open source* via *GitHub* i els podem trobar en el següent enllaç on també podem trobar els generadors de *datasets*:

https://github.com/VictorSuarezVara/Reading-analog-clocks-with-neural-networks

També podem visualitzar algunes de les execucions més significatives realitzades via *Wandb* (l'eina utilitzada per visualitzar execucions en el següent enllaç:

https://wandb.ai/dowhiler/pytorch-demo

A mode personal aquest procés ha servit per aprendre a manejar xarxes neuronals utilitzant eines punteres i descobrir incomptables detalls del funcionament d'aquesta tecnologia fins al punt de sentir-me capaç de portar un projecte de xarxes neuronals en solitari ja que ara sé el que puc, i el que no puc fer, amb xarxes neuronals.

Finalment hem donat amb una primera solució per a la lectura de qualsevol rellotge analògic: amb el programa que, a partir de les manetes i la circumferència d'un rellotge real, crea un *dataset* amb el qual entrenant una xarxa neuronal aconseguim llegir l'hora de qualsevol imatge frontal d'aquest mateix rellotge.

9 Conclusions

Hem complert tots els objectius satisfactòriament. La idea inicial que perseguia el projecte hem aconseguit i ha estat una recerca plena de descobriments i aprenentatge.

Hem acabat amb múltiples contribucions realitzades per la lectura de rellotges usant xarxes neuronals i, tot i que si ara tornéssim a realitzar el projecte canviaríem el 90% de les decisions que hem pres, ens sentim preparats per començar una nova recerca usant xarxes neuronals.

Pel que fa a les xarxes neuronals, acabem el treball amb la idea que es fonamenten en aprendre del que veuen i el lliure albir.

Hagués aportat molt a la investigació expandir el problema a la lectura d'altres instruments analògics, i veure si el model pot captar també altres instruments. De totes formes, ha estat molt enriquidor seguir la línia que ens marcava l'*experiment design*.

Pel que fa a la problemàtica de relacionar text i formes geomètriques, vist en perspectiva, hem aconseguit que una xarxa neuronal llegeixi imatges on es dona aquesta relació però no fent aquesta relació. Caldria un experiment per acabar de refutar que efectivament la xarxa no està realitzant aquesta relació per a determinar l'hora sinó que està seleccionat només uns píxels, però tot apunta cap a aquesta conclusió.

Proposem ara futures noves línies d'investigació tot i que no estem en posició de proposar-ho amb seguretat, ja que encara ens queda molt per saber sobre xarxes neuronals.

Per fer que la xarxa neuronal aprengui a captar els elements que volem que relacioni proposem dues línies de treball:

Entrenar la xarxa donant-li sempre un rellotge que mai

ha vist abans. Això ho podríem realitzar fent que el *Dataloader* fos en si mateix el generador d'imatges de rellotges que hem creat o trobant la manera de crear imatges de rellotges falses amb un altre xarxa neuronal.

Un altre via seria afegir una *loss* a la meitat de l'arquitectura que determines si s'està reconeixent les parts significatives per fer la relació (les manetes i els números) o no. Així forçaríem a la xarxa neuronal a aprendre del que sabem que es important.

Així doncs, donem per finalitzada la recerca aprenent a llegir l'hora amb xarxes neuronals, realitzada durant 5 mesos amb col·laboració amb el CVC.

AGRAÏMENTS

Al tutor d'aquest Treball de Final de Grau, Dimosthenis Karatzas per les interminables xerrades sobre intel·ligència i xarxes neuronals i per la llibertat dipositada en mi. Als meus pares i avis per finançar-me la carrera i oferir-me la motivació. Al CVC per oferir-me la possibilitat de realitzar aquesta recerca per ells i, per últim, als estudiants i investigadors del CVC, per deixar-me un lloc a la taula en cada dinar.

Bibliografia

[1] R. Sablatnig and Walter G. Kropatsch. Automatic reading of Analog Display Instruments. Technical University Vienna, 1994.

[2] Han Jiale, Li En, Tao Bingjie and Lv Ming. Reading Recognition Method of Analog Measuring Instruments Based on Improved Hough Transform. In University of China, 2011.

[3] Shiva Verma. Reading Clocks using Neural Nets. [en línia]. (24 de gener, 2020). Recuperat de https://towardsdatascience.com/training-neural-net-to-read-clock-time-9473175171e3

[4] K. He, X. Zhang, S. Ren and J. SunDeep. Residual Learning for Image Recognition. Microsoft Research, 2015.

[5] R. Hu, A. Singh, T. Darrell, and M. Rohrbach. Iterative Answer Prediction with Pointer-Augmented Multimodal Transformers for TextVQA. Facebook (FAIR) and University of California, Berkeley, 2020.

[6] Felix Duvallet. Deep time (reading clocks using PyTorch) [en línia]. (9 d'octubre de 2016). Recuperat de https://felixduvallet.github.io/blog/deep.time/#header-generate-clocks

[7] Arjun Singh Saud and Subarna Shakya. Analysis of l2 regularization hyperparameter for stock price prediction. University, Kathmandu, Nepal. 2021

[8] DeepLearning.AI. Coursera. Programa especializado: Aprendizaje profundo. [en línia]. [consultat: 10 maig 2021]. Recuperat de https://www.coursera.org/specializations/deep-learning#courses

[9] Marcos Merino. La Paradoja de Moravec: por qué la inteligencia artificial hace fácil lo difícil (y viceversa). Inteligencia artificial. [en línia]. (5 d'agost, 2019) Recuperat de https://www.xataka.com/inteligencia-artificial/paradoja-moravec-que-inteligencia-artificial-hace-facil-dificil-viceversa

[10] Michael A. Nielsen, "Neural Networks and Deep Learning", Determination Press, 2015. Chapter 4: A visual proof that neural nets can compute any function [Recurs electrònic]. [consultat: 17 febrer 2021]. Recuperat de: http://neuralnetworksanddeeplearning.com/chap4.html

[11] Manifiesto por el Desarrollo Àgil de Software [en línia]. (2001). Recuperat de http://agilemanifesto.org/iso/es/manifesto.html

[12] Analog-Clocks. Shiva Verma, [en línia]. (24 de juliol, 2020). Recuperat de https://www.kaggle.com/shivajbd/analog-clocks

[13] Combine. Reading an analog clock using image processing. [en línia]. (4 de septembre de 2018). Recuperat de https://combine.se/reading-an-analog-clock-using-image-processing/

[14] A. Quattoni, and A.Torralba. Recognizing Indoor Scenes. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

[15] Matthew D. Zeiler, and R. Fergus. Visualizing and Understanding Convolutional Networks. In CVPR, 2013.