

Millora de la seguretat a l'escola amb intel·ligència artificial

Autor: Pau Sevilla Rodríguez

Tutor: Josep M. Zambudio Cantón

Curs: 2n BTX A

Centre: Maristes Rubí

Localitat i data: A Rubí el 26 octubre de 2022

ÍNDEX

INTRODUCCIÓ.....	6
MARC TEÒRIC.....	7
1 HISTÒRIA DE LA INTEL·LIGÈNCIA ARTIFICIAL.....	8
1.1 Màquina de Turing i naixement de la informàtica.....	9
1.1.1 Màquina de Turing.....	9
1.1.2 L'arquitectura de John von Neumann.....	10
1.2 Conferència de Dartmouth	11
1.3 Els anys d'or.....	12
1.3.1 Student (Daniel Bobrow).....	12
1.3.2 Eliza (Xat bot).....	12
1.4 El primer hivern de la IA	13
1.5 El boom.....	14
1.6 El segon hivern de la IA.....	14
1.7 Actualitat.....	14
2 CONCEPTES I FUNCIONAMENT.....	15
2.1 Intel·ligència artificial	15
2.1.1 Tipus D'IA.....	15
2.1.2 Machine Learning	16
2.1.2.1 Overfitting i Underfitting.....	16
2.1.3 Deep Learning.....	17
2.1.4 Big Data.....	17
2.1.5 Tipus d'aprenentatge	18
2.2 Xarxes neuronals	19
2.2.1 Xarxes neuronals biològiques.....	19

2.2.2	Comportament de la neurona	20
2.2.3	Parts d'una neurona artificial	20
2.2.4	Funcions d'activació	22
2.2.5	Parts d'una Xarxa Neuronal	24
2.2.6	Algoritmes d'aprenentatge de les xarxes neuronals	24
2.2.7	Models	25
2.2.8	Regressió lineal	25
2.2.9	Error del model	26
2.2.9.1	Mínim quadrats ordinaris	26
2.2.9.2	Descens del gradient	27
2.2.9.3	Algoritme de backpropagation.....	29
2.2.10	Etaques de la xarxa neuronal	31
3	INTEL·LIGÈNCIA ARTIFICIAL APLICADA AL RECONeixEMENT D'IMATGES.....	32
3.1	Actualitat del reconeixement d'imatges	32
3.2	Xarxes neuronals convolucionals	33
3.2.1	Patrons en la xarxa convolucional	34
3.2.2	Filtres en la xarxa convolucional.....	34
3.2.3	Convolucions.....	35
3.2.4	Pooling	36
3.2.5	Dropout	37
4	ÈTICA DE LA IA	37
	MARC PRÀCTIC	39
1	INTRODUCCIÓ A LA PART PRÀCTICA	40
1.1	Origen de la idea	40
1.2	Explicació de la Idea	40
1.3	Passos seguits.....	41

2	PROGRAMACIÓ ALGORITME IA	42
2.1	Justificació llenguatge de programació i llibreries utilitzades	42
2.2	Explicació de l'algoritme	43
2.2.1	Explicació de l'arxiu 'main.py'	44
2.2.2	Explicació del arxiu 'simple_facerec.py'	46
2.2.3	Resultats del algoritme	48
2.3	Problemes i solucions trobades	49
2.3.1	Problema amb model obsolet	49
2.3.2	Problema amb la falta d'imatges	50
2.3.3	Error al detectar una cara en un frame	51
2.3.4	Creació de lag en el vídeo al enviar el log a la BD (Base de dades)	52
3	PROGRAMACIÓ D'UNA PÀGINA WEB	53
3.1	Justificació de les llibreries i llenguatge utilitzat	53
3.2	Esquemes d'organització de la pàgina web	54
3.3	Creació d'una base de dades	55
3.3.1	Taules de la db (Base de Dades)	55
3.4	Funcionament i dissenys de la pàgina web	57
4	RESULTAT FINAL	59
5	POSSIBLES MILLORES	60
	CONCLUSIONS	61
	BIBLIOGRAFIA	62
	ANNEX	66
	Part Python	66
	Algoritme de detecció de rostres	66
	Algoritme d'augment d'imatges	68
	Arxiu amb els mètodes de detecció de rostres.	69

Part Flutter	71
Arxiu Main	71
Arxiu pàgina inicial.....	72
Arxiu llista d'alumnes.....	76
Arxiu de rutes	84
AGRAÏMENTS.....	85

INTRODUCCIÓ

En els temps que corren avui en dia, qui no ha sentit parlar de cotxes autònoms, assistents virtuals, robots que poden mantenir converses amb les persones, etc? Tots aquests temes tenen una cosa en comú i és que són algoritmes d'intel·ligència artificial.

Per començar, la hipòtesi que he volgut formular en el meu treball de recerca ha estat: *“La intel·ligència artificial pot millorar la seguretat a l'escola”*. En un principi, tenia pensat intentar demostrar que la IA pot ajudar a l'aula, ja sigui als professors o a l'alumnat, però investigant maneres d'aplicar aquests algoritmes no en vaig trobar cap que pogués demostrar una ajuda clara, així que vaig començar a pensar en altres hipòtesis.

Se'm va ocórrer preguntar a coneguts que treballen a col·legis si saben problemes que creien que podien solucionar amb algun algoritme de IA. Llavors, un d'ells em va dir que a les sortides dels nens petits del col·legi, el professorat havia d'estar pendent de donar cada nen/a al veritable pare/mare, cosa força difícil sobretot a principi de curs. Així doncs, vaig pensar que amb una IA de reconeixement facial podria comprovar si els pares estan autoritzats o no a recollir el nen o nena i comunicar-li-ho al professor que estigués a la porta. Un altre exemple que vaig pensar és que si fiquen a la base de dades les cares del professorat, amb el mateix programa la IA podria detectar qui es cola a la sala de professors o de material i, d'aquesta manera, millorar la seguretat de l'escola.

El meu pare és informàtic i ja des de petit m'ha intentat enganxar a aquest magnífic món. Finalment, ho va aconseguir i jo, cada any, intentava aprendre algun concepte nou. Justament aquest any em volia ficar en el món de la IA i, en veure que havia de fer el TDR, vaig pensar que seria l'excusa perfecta per aprendre intel·ligència artificial. És un món que m'atrau bastant, ja que m'encanten les matemàtiques, però també m'encanta la informàtica i justament la IA és la unió de les dues matèries.

MARC

TEÒRIC

1 HISTÒRIA DE LA INTEL·LIGÈNCIA ARTIFICIAL

La intel·ligència artificial ha estat present al nostre món des de fa molt temps, com a mínim la idea. Posar-ho a la pràctica va ser més difícil perquè les idees es van avançar a la tecnologia i en l'època on no hi havia ordinadors potents, que és quan va sorgir el dubte de si els ordinadors podien pensar, era impossible provar aquestes teories.

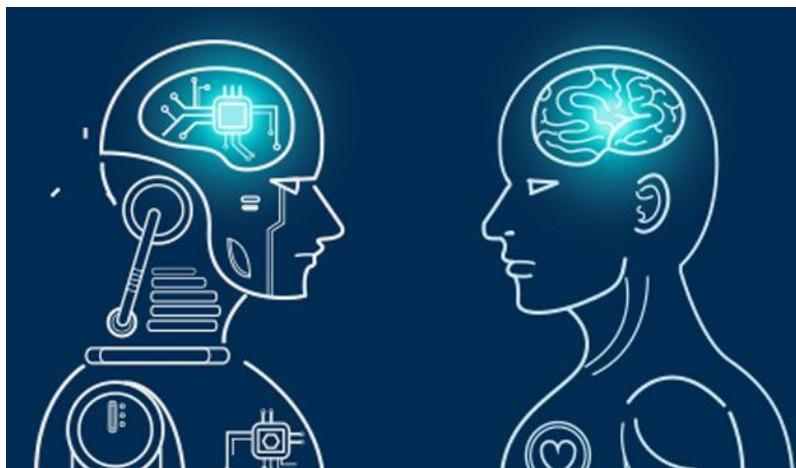


Fig. 1: Humà i IA

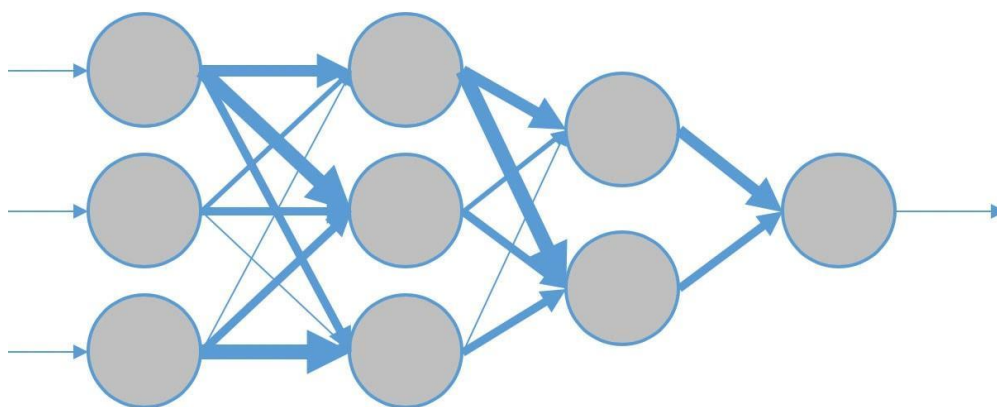


Fig. 2: Xarxa neuronal

1.1 Màquina de Turing i naixement de la informàtica

Per començar, si es vol saber i comprendre l'inici de la intel·ligència artificial, un s'ha de remuntar a l'inici de la informàtica. Justament al començament de la Segona Guerra Mundial.

Alemanya tenia una màquina per poder encriptar i desencriptar missatges alemanys anomenada Enigma. La màquina consistia en una sèrie de rotors, concretament 5, que permetien canviar el sistema d'encriptació cada vegada que s'escrivia amb tan sols una lletra. Això generava trillions de possibilitats i com que el codi canviava cada dia era impossible de desencriptar manualment. Alan Turing va desenvolupar la Màquina, i per això porta el seu cognom. Bàsicament era el primer ordinador creat capaç de fer càlculs matemàtics molt més ràpid que els humans. Gràcies a això la Segona Guerra Mundial va durar dos anys menys.



Fig. 3: Alan Turing

1.1.1 Màquina de Turing

La definició tècnica de la màquina de Turing seria que es tracta d'un model matemàtic que consisteix en un autòmat capaç d'implementar qualsevol problema matemàtic expressat a través d'un algoritme. Té un gran valor i potencial ja que es pot adaptar a qualsevol algoritme.



Fig. 4: Màquina de Turing

La màquina de Turing consta d'una cinta de paper, un capçal mòbil que és capaç de realitzar traces i un processador central. La cinta, en terminologia informàtica, seria la memòria on s'emmagatzemen els 0 i 1 de l'ordinador. El capçal té la funció de gravar un traç a la cinta i el processador serveix per emmagatzemar el codi i els algorismes matemàtics.

L'objectiu final de la màquina de Turing és el gravat de símbols i números que poden ser des de codi binari fins a números. Per organitzar els canvis d'escriptura hi ha els estats Q els quals pot programar l'usuari. Finalment, també existeix una funció on es trobarien el procés matemàtic, els canvis d'estat, la lectura de les cel·les... Com es pot veure hi ha una gran semblança entre la màquina de Turing i els ordinadors d'avui en dia.

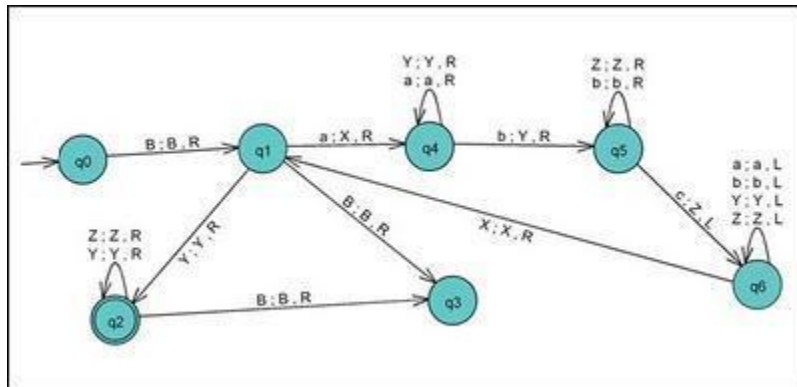


Fig. 5: Estats Q Màquina Turing

1.1.2 L'arquitectura de John von Neumann

John von Neumann va néixer a Budapest el 28 de desembre de 1903. Va ser una persona excel·lent en els estudis i ràpidament va destacar. Quan va començar a ser professor va conèixer a Alan Turing mentre aquest feia el seu doctorat; allà van forjar una gran amistat i van compartir la seva gran passió per les matemàtiques i la computació. Deu anys més tard, després que Turing publicés la seva tesi sobre la màquina de Turing, Von Neumann va idear una arquitectura (fabricació d'un sistema informàtic) que és la que s'utilitza avui en dia en els ordinadors actuals.



Fig. 6: John Von Neumann

Aquesta arquitectura proposa la utilització d'una unitat central de processament o CPU, que conté una unitat aritmètica lògica capaç de fer càlculs matemàtics senzills, un conjunt de registres que permeten l'emmagatzematge temporal de dades i adreces de memòria, i una unitat de control que s'encarrega de recollir les instruccions des de la memòria principal, de descodificar-les i executar-les. A més, aquest model d'arquitectura també suggereix l'existència d'una memòria principal on resideixen les dades i les instruccions, i d'un bus d'entrada i sortida que permet carregar els programes i les dades des d'un mitjà d'emmagatzematge extern i lliurar-ne un resultat .

1.2 Conferència de Dartmouth

La conferència de Dartmouth va tenir lloc l'estiu de 1956 i va ser organitzada per Marvin Minsk, John McCarthy i Claude Shannon. Allà es va reunir a tots aquells científics que treballaven en el camp de la intel·ligència artificial. Un dels primers resultats que va sorgir d'aquesta conferència va ser l'expressió 'intel·ligència artificial' per a tots els termes d'aquest àmbit. Alguns dels problemes que es van tractar de resoldre en aquesta conferència van ser: com es pot programar un ordinador amb un llenguatge de programació, xarxes neuronals, teories de càlculs, etc.



Fig. 7: Integrants conferència de Dartmouth

1.3 Els anys d'or

Els anys cinquanta del segle XX van ser, sens dubte, dels més interessants pel que fa al desenvolupament de la intel·ligència artificial. Els inversors van veure la idea clara i tothom va començar a invertir en projectes d'intel·ligència artificial. Això va provocar que aquesta branca creixés molt ràpidament. Es van crear programes increïbles com Student de Daniel Bobrow o Eliza, el primer xat bot de la història. També es va aprofundir en xarxes semàntiques, sobretot de traducció del llenguatge natural.

1.3.1 Student (Daniel Bobrow)

Student és un dels primers programes d'intel·ligència artificial que resol problemes d'àlgebra. Va ser escrit en Lisp per Daniel G. Bobrow com a tesi doctoral l'any 1964 i fou dissenyat per llegir i resoldre el tipus de qüestions que es troben als llibres d'àlgebra de secundària. El programa és un dels primers assoliments de la IA en el processament del llenguatge natural.



Fig. 8: Daniel G. Bobrow

1.3.2 Eliza (Xat bot)

Joseph Weizenbaum, partint de la idea que les màquines són capaces de comportar-se com els humans, el 1966 va desenvolupar un programa amb què pretenia ser capaç d'enganyar els humans fent-los pensar que estaven parlant amb una persona. Aquest bot es deia ELIZA i va ser dissenyat com un mètode per mostrar la



Fig. 9: Conversació amb Eliza

superficialitat de la comunicació entre l'home i la màquina. Ho feia reconeixent paraules clau i preguntant-los coses com si fos un psicòleg. Per exemple, si algú esmentava la mare en una frase, el bot automàticament li demanaria que li

digués més informació sobre la família. D'aquesta manera, es creava una il·lusió d'interacció real.

1.4 El primer hivern de la IA

Aquest temps de progrés no va durar molt perquè com que s'hi invertien tants diners però no es generaven els mateixos amb els resultats, es va crear una bombolla i ràpidament els inversors van deixar de posar diners. A més a més, van sorgir diverses persones que defensaven que la IA era un camp sense futur i una pèrdua de temps. I així es va entrar en el primer hivern de la IA. Durant aquesta etapa es van trobar tres grans problemes:

1. **Falta de potència:** Els ordinadors de l'època -estem parlant de 1970- tenien 4325 cops menys potència que un mòbil actual d'un valor de cent euros en el mercat. Un exemple d'aquesta falta de potència es pot observar en la creació d'un traductor de rus a anglès que trigava deu minuts a traduir i només es podien introduir vint paraules.
2. **Falta de dades:** Perquè un ordinador pugui, per exemple, reconèixer cares es necessita una base de dades amb estructures facials, mides d'ulls, colors d'ulls i moltes més dades d'informació. Tota aquesta informació no cabia en els ordinadors de l'època, ja que no hi havia tant d'espai.
3. **Falta de recursos:** En Tots els algoritmes d'IA, el seu grau d'eficiència va amb temps exponencial, és a dir, contra més temps i recursos s'hi dediquin millor aniran i seran més efectius. Llavors, els algoritmes podien resoldre problemes de "joguina"; per exemple, podien jugar als escacs, però en un taulell de 2x2 ja que, malgrat que puguin jugar en qualsevol taulell d'escacs, no tenien els recursos necessaris.



Fig. 10: Ordinador de l'època

1.5 El boom

Al voltant de 1980 van sorgir diversos conceptes que van revolucionar al món de la informàtica i, d'aquesta manera, facilitaven molt el desenvolupament de la IA. Entre d'altres, els més importants són els Frames, les classes i les herències. A més a més, a part d'això es va crear un nou tipus de programes: els programes experts. Aquests emmagatzemaven conceptes de professionals i resolien dubtes de les empreses. Això va atreure les empreses, ja que els eren útils. Per exemple, l'empresa Digital Systems va estimar que estalviava fins a quaranta milions de dòlars anuals per l'ús de XCON, un sistema expert per ajudar a muntar ordinadors.

1.6 El segon hivern de la IA

Però, com sempre, les històries es repeteixen i una vegada més, per l'eufòria de les novetats i les especulacions, es va crear una bombolla que quan va explotar va provocar que unes 300 empreses d'intel·ligència artificial se n'anessin a la bancarrota i els sistemes experts es desactualitzessin fins a ser inservibles. Malgrat això, van seguir els estudis en aquest àmbit i es van aconseguir fites increïbles com pot ser el Deep Blue, la primera IA a guanyar una partida d'escacs al millor jugador del món.

1.7 Actualitat

Actualment, la manca d'espai ja no existeix donat que s'han creat en pocs anys aparells mil vegades més potents, i la falta de dades es va acabar amb el naixement d'internet ja que ara hi ha accés a trillions de dades gratis. Això ha fet augmentar la velocitat d'evolució molt ràpidament, s'han creat mil algoritmes d'IA i eines per crear-la amb més facilitat. La majoria d'empreses grans tenen un departament d'intel·ligència artificial per ajudar a millorar la qualitat del producte que ofereixen i també s'han creat assistents virtuals, cotxes autònoms... Però la IA encara és molt bàsica i el que pot fer ara mateix són tasques senzilles. De fet, encara no se sap com es pot crear una IA capaç de tenir més de dues funcions.

2 CONCEPTES I FUNCIONAMENT

2.1 Intel·ligència artificial

La intel·ligència artificial és el camp de la informàtica que busca la creació de màquines que puguin **imitar comportaments intel·ligents**, com per exemple conduir, reconèixer veus, escriure, etc. Un sistema d'IA el que fa és analitzar grans grups de dades (Big Data) i identificar patrons o tendències per després poder formular prediccions de forma automàtica, ràpida i amb precisió.

2.1.1 Tipus D'IA

Dins de la intel·ligència artificial, segons el que aquesta pugui fer i el nombre de tasques que pugui realitzar hi ha dos tipus d'IA:

- **IA forta:** Aquesta pot aprendre a realitzar qualsevol tasca que realitzi un ésser humà.
- **IA dèbil:** Aquesta es limita a unes poques tasques específiques. Només sap el que li ensenyen. Per exemple, si l'entrenen per reconèixer tumors a les radiografies i una d'aquesta radiografies està girada, no reconeixerà la radiografia ja que no li han ensenyat que les radiografies poden estar girades. En canvi, a un humà quan li ensenyen una vegada com és una radiografia la pot identificar de qualsevol manera.

Avui en dia, a l'únic lloc on existeix la intel·ligència artificial forta és a les pel·lícules perquè els investigadors encara no saben com crear-la. Malgrat que hagin intentat imitar el comportament del cervell humà només han aconseguit crear IA dèbil.

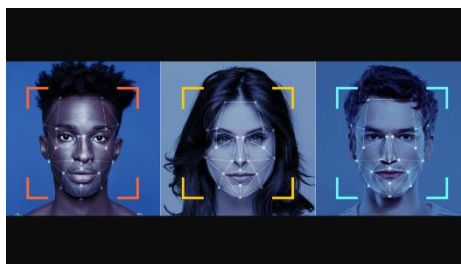


Fig. 11: Exemple IA dèbil

2.1.2 Machine Learning

Dins de la IA hi ha diverses agrupacions, però el més utilitzat per tots és el **Machine learning** que significa dotar a les màquines d'aprenentatge. Bàsicament, l'aprenentatge automàtic utilitza algorismes per analitzar dades, aprendre i, finalment, poder fer una predicció o suggeriment sobre alguna cosa. El procés d'aprenentatge comença amb observacions, dades, experiències o instruccions que s'analitzen per trobar patrons i, d'aquesta manera, poder prendre millors decisions sobre el futur basades en els exemples que s'ofereixen. Com abans a la IA hi ha diverses formes d'aconseguir aquest objectiu, però la més utilitzada és la de les xarxes neuronals (en la pag. queda explicat). Quan s'usen moltes dades i la xarxa neuronal és molt gran s'anomena aprenentatge profund (**deep learning**) el qual destaca per ser molt complex.

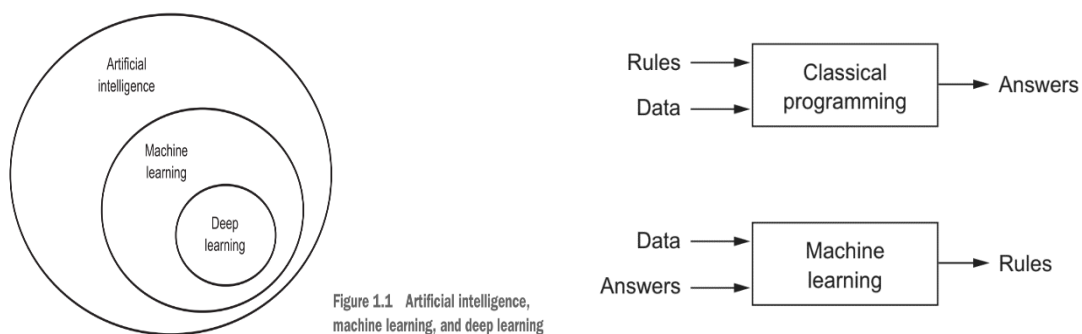


Fig. 12: Machine learning vs classical programming // Machine learning, AI, Deep learning

2.1.2.1 Overfitting i Underfitting

L'overfitting es produeix quan una IA està tan entrenada que només sap resoldre i funcionar amb les dades amb les que se l'ha entrenat. Si poguéssim veure la corba d'aprenentatge d'un algoritme es podria veure com cada vegada l'error va descendent però, de sobte, s'estanca i comença a pujar; just en aquest moment és quan s'està generant *overfitting* i l'algoritme està sobreentrenat.

Per altra banda, **l'underfitting** és tot el contrari, ja que es dona quan un algoritme d'IA fallarà perquè li falten dades per poder funcionar en tots els casos. Per exemplificar tot això, es podria dir que un algoritme ben entrenat seria la persona que estudia dia a dia per l'examen i entén els conceptes.

L'*overfitting* seria aquell alumne que només s'estudia els exercicis de classe, però no entén el temari i, per això, amb el mínim canvi falla. I l'*underfitting* seria aquell que directament no estudia.

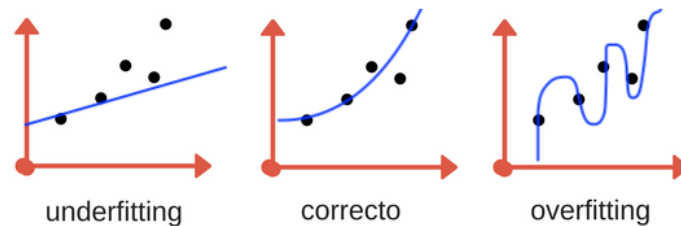


Fig. 13: Gràfica exemple overfitting, underfitting

2.1.3 Deep Learning

El *Deep Learning* conegut també com a “aprenentatge profund” és un subapartat de l'aprenentatge automàtic que se centra en algorismes basats en xarxes neuronals. L'aprenentatge profund ha fet néixer un tipus d'aprenentatge on es prioritza l'aprenentatge per capes i on cada capa a mesura que avança serà més gran i més complexa. Com més gran sigui la xarxa millor seran els resultats. És a dir, el *deep learning* és una xarxa neuronal però molt més gran i complexa.

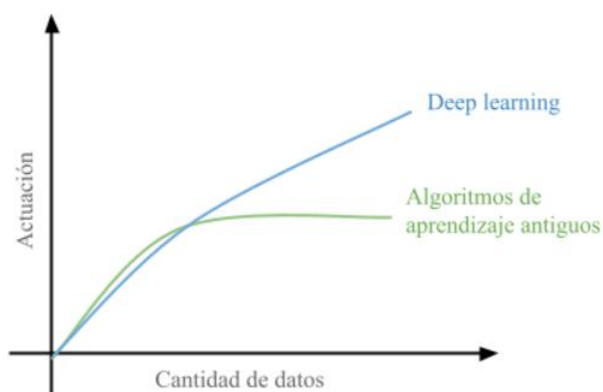


Fig. 14: Gràfica comparació deep learning vs altres algorismes

2.1.4 Big Data

El *Big data*, també anomenat “dades massives”, és el terme que s'utilitza per a referir-se a una quantitat de dades molt gran. Amb la creació d'internet el 1983,

quan es van començar a acumular dades i dades és va facilitar molt la creació de IA i avui en dia gràcies a tot aquest conjunt de dades s'han pogut crear eines per controlar-les i usar-les. L'ús d'aquestes en el camp de la IA ha fet millorar molt els algoritmes.



Fig. 15: Big Data

2.1.5 Tipus d'aprenentatge

A un algoritme d'intel·ligència artificial se'l pot ensenyar de diverses maneres, però sobretot existeixen dues formes de fer-ho:

- **Aprenentatge supervisat:** En aquest cas, s'ha de crear una relació entre les dades d'entrada i les de sortida, és a dir, que d'una certa manera s'ha de participar en la supervisió de l'aprenentatge. Per exemple, per ensenyar un algoritme a reconèixer gats i gossos se li donen moltes imatges de gossos i gats i a cada una d'aquestes se l'etiqueta com a gats o gossos. Quan l'algoritme hagi revisat milers d'imatges podrà dir d'una de nova si és un gat o un gos.
- **Aprenentatge no supervisat:** L'algoritme en aquest cas aconsegueix coneixement amb només dades d'entrada sinó que normalment el que fa és trobar les relacions entre les dades d'entrada i treure'n un resultat. Aquesta manera s'utilitza per problemes de classificació però no és molt comú ja que gairebé sempre s'utilitza l'aprenentatge supervisat.

2.2 Xarxes neuronals

El cervell humà és el sistema de càlcul més complex que l'home coneix. De fet, la capacitat del cervell humà de pensar, recordar i resoldre problemes ha inspirat a molts científics per intentar copiar a l'ordinador el funcionament del cervell humà. Però l'ordinador i l'home realitzen diferents classes de tasques i cadascú és millor en una o altra. Així com l'operació de reconèixer el rostre d'una persona resulta una tasca relativament senzilla per a l'home i difícil per a l'ordinador, la comptabilitat d'una empresa és una tasca costosa per a un expert comptable i una senzilla rutina per a un ordinador bàsic.

Les Xarxes Neuronals Artificials, ANN (*Artificial Neural Networks*) estan inspirades en les xarxes neuronals biològiques del cervell humà. Estan constituïdes per elements que es comporten de manera similar a la neurona biològica en les seves funcions més comunes i els seus elements estan organitzats de manera semblant a la que presenta el cervell humà.

2.2.1 Xarxes neuronals biològiques

El cervell humà funciona internament a partir de neurones i de xarxes neuronals conegudes com a xarxes neuronals biològiques. A un nivell molt elevat, les neurones interactuen i es comuniquen entre elles mitjançant el nexse d'unió entre els terminals d'axons que estan connectats a les dendrites mitjançant una bretxa.

En un sentit més matemàtic, quan una neurona envia un missatge a una altra mitjançant aquesta xarxa només si la suma ponderada dels senyals d'entrada d'una o més neurones és prou gran es transmetrà el missatge. Es parla d'activació quan se supera el llindar i el missatge es transmet a la neurona següent.

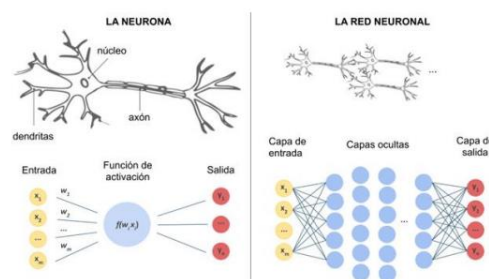


Fig. 16: Xarxa Neuronal Biològica vs artificial

2.2.2 Comportament de la neurona

Les ANN -al marge d' "assemblar-se" al cervell- presenten una sèrie de característiques pròpies del cervell:

- 1 Aprendre:** adquirir el coneixement d'un concepte per mitjà de l'estudi, l'exercici o l'experiència. Les ANN poden canviar el comportament en funció de l'entorn ja que quan se'ls mostra un conjunt d'entrades, s'ajusten elles mateixes per produir unes sortides consistents.
- 2 Generalitzar:** estendre o ampliar una cosa. Les ANN generalitzen automàticament a causa de la seva pròpia estructura i naturalesa. Aquestes xarxes poden oferir -dins un marge- respostes correctes a entrades que presenten petites variacions a causa dels efectes de soroll o de distorsió.
- 3 Abstraure:** aïllar mentalment o considerar per separat les qualitats d'un objecte. Algunes ANN són capaces d'abstreure l'essència d'un conjunt d'entrades que aparentment no presenten aspectes comuns o relatius.

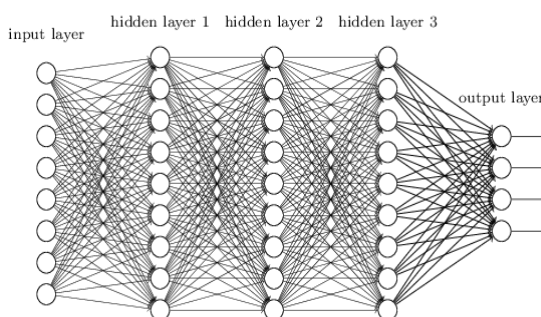


Fig. 17: Xarxa Neuronal

2.2.3 Parts d'una neurona artificial

Una xarxa neuronal està composta per tres parts: entrada, nucli i sortides.

Les entrades reben les dades o paràmetres que permeten que les neurones decideixin si estar actives o no. La seva nomenclatura és: X_1, X_2, \dots, X_n

Entre l'entrada i el nucli hi ha els pesos (W_1, W_2, \dots, W_n), que representen la memòria de la xarxa.

Al nucli es fan totes les operacions necessàries per determinar la sortida de la neurona; el procés que es realitza al nucli varia depenent de la xarxa neuronal que s'estigui treballant.

Les sortides tornen la resposta de la neurona, és a dir, si està activa o no, representades comunament com Y_1, Y_2, \dots, Y_n .

Al nucli es realitzen 3 tipus d'operacions per determinar la sortida de la neurona que són: Regla de propagació, Funció d'activació i Funció de sortida.

La regla de propagació, integra la informació provinent de les diferents neurones artificials i proporciona el valor del potencial postsinàptic de la neurona. La funció d'activació, proveeix l'estat d'activació actual de la neurona. Per últim, la funció de sortida representa la sortida actual de la neurona.

Per aprofundir més en els conceptes anteriors:

Entrades i sortides

Les entrades i sortides d'una neurona es poden classificar en dos grans grups: binàries o contínues. Les neurones binàries només admeten dos valors possibles. En general, en aquest tipus de neurona s'utilitzen els dos alfabetes $\{0,1\}$ o $\{-1,1\}$. Per altra banda, les neurones contínues admeten valors dins d'un determinat rang, que en general solen definir-se com a $[-1, 1]$. La selecció del tipus de neurona a usar depèn de l'aplicació i del model a construir.

Pesos

El pes sinàptic w_{ij} defineix la força d'una connexió sinàptica entre dues neurones, la neurona presinàptica i i la neurona postsinàptica j . Els pesos sinàptics poden tenir valors positius, negatius o zero. En cas d'una entrada positiva, un pes positiu actua com a excitador, mentre que un pes negatiu actua com a inhibidor. En cas que el pes sigui zero, no hi ha comunicació entre el parell de neurones. Mitjançant l'ajust dels pesos sinàptics, la xarxa és capaç d'adaptar-se a qualsevol entorn i executar una tasca determinada.

Regla de programació

La regla de programació determina el resultant potencial de la interacció de la neurona amb les N neurones veïnes. La més simple i utilitzada consisteix a fer una suma ponderada de les entrades amb els seus pesos corresponents:

$$net_i(t) = \sum_{j=1}^N W_{ij} * X_j(t)$$

Funció de sortida

La funció de sortida proporciona el valor de sortida de la neurona, segons l'estat d'activació de la neurona. És a dir:

$$y_i(t) = f(net_i(t))$$

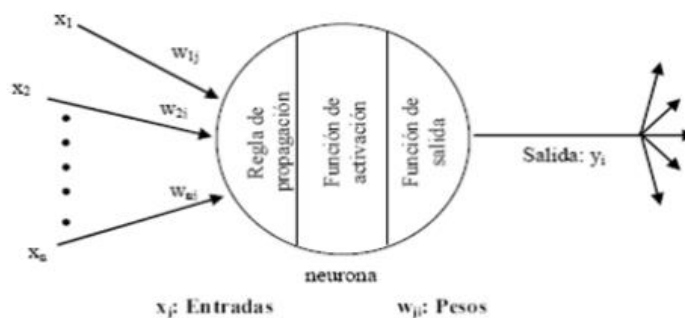


Fig. 18: Neurones i les seves parts

2.2.4 Funcions d'activació

La funció d'activació d'una xarxa neuronal és bàsicament la transformació de l'entrada de les capes mitjançant una suma ponderada de la sortida d'aquestes. Aquesta operació dona un número entre els valors de zero a u. Això serveix per trencar la linealitat de la xarxa ja que fer una suma ponderada de neurones (cada neurona és una regressió lineal) és equivalent a fer només una operació. És per això que s'ha de trencar la linealitat. N'hi ha diferents tipus:

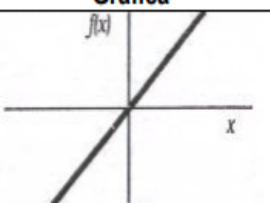
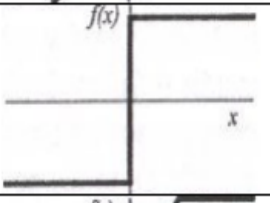
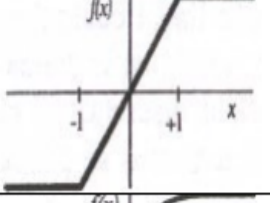
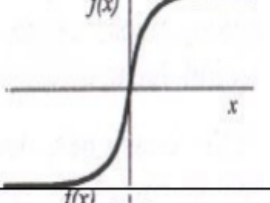
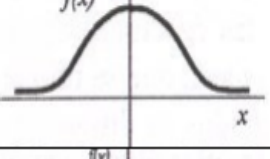
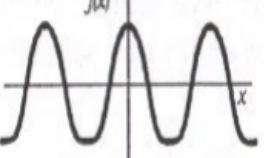
**** Observació ****

La funció d'activació determina l'estat d'activació actual de la neurona sobre la base del potencial resultant net_i i l'estat d'activació anterior de la neurona $a_i(t - 1)$. L'estat d'activació de la neurona per a un determinat instant de temps(t) pot ser expressat de la manera següent:

$$a_i(t) = f(a_i(t - 1), net_i(t))$$

Tanmateix, a la majoria dels models se sol ignorar l'estat anterior de la neurona, definint-se l'estat d'activació en funció del potencial resultant.

$$a_i(t) = f(net_i(t))$$

	Función	Rango	Gráfica
Identidad	$y = x$	$[-\infty, \infty]$	
Escalón	$y = \begin{cases} 1, & \text{si } x \geq 0 \\ 0, & \text{si } x < 0 \end{cases}$ $y = \begin{cases} 1, & \text{si } x \geq 0 \\ -1, & \text{si } x < 0 \end{cases}$	$[0, 1]$ $[-1, 1]$	
Lineal a tramos	$y = \begin{cases} 1, & \text{si } x > 1 \\ x, & \text{si } -1 \leq x \leq 1 \\ -1, & \text{si } x < -1 \end{cases}$	$[-1, 1]$	
Sigmoidea	$y = \frac{1}{1 + e^{-x}}$ $y = \tanh(x)$	$[0, 1]$ $[-1, 1]$	
Gaussiana	$y = Ae^{-Bx^2}$	$[0, 1]$	
Sinusoidal	$y = A \sin(wx + \varphi)$	$[-1, 1]$	

2.2.5 Parts d'una Xarxa Neuronal

Les xarxes neuronals es poden dividir en tres parts: la **capa d'entrada** on es recullen els inputs i s'utilitzen en els processos de càlcul; les **capes ocultes** que són les que hi ha entre la de sortida i la d'entrada i que recullen els valors de l'anterior i els passen a la següent i, finalment, la **capa de sortida** que és l'encarregada de donar el resultat. A cada capa hi ha les neurones necessàries per resoldre el problema.

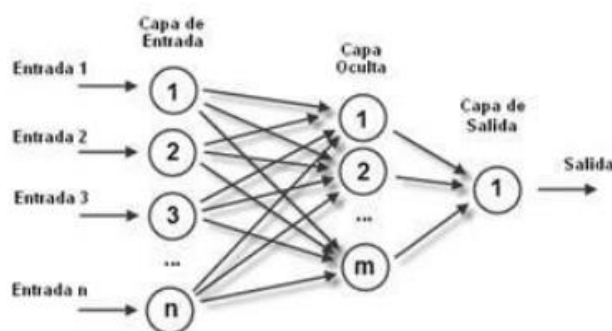


Fig. 19: Xarxa neuronal i les seves parts

2.2.6 Algoritmes d'aprenentatge de les xarxes neuronals

En aprenentatge profund, els algorismes d'aprenentatge són un reflex de com el cervell humà analitza informació i l'aprèn. Quan un model s'entrena, els algorismes se centren a analitzar els elements desconeguts dels senyals d'entrada per extreure'n característiques, relacionar conceptes i descobrir patrons de dades útils. El model d'aprenentatge profund utilitza diferents algorismes. Depenent del problema que es vulgui solucionar, hi haurà alguns algorismes més adequats. Aquests són els 10 algorismes més utilitzats:

1. Xarxes neuronals convolucionals (CNN)
2. Xarxes de memòria a curt termini (LSTM)
3. Xarxes neuronals recurrents (RNN)
4. Xarxes generatives adversàries (GANs)
5. Xarxes de funcions de base radial (RBFN)
6. Perceptrons multicapa (MLP)
7. Mapes autoorganitzatius (SOM)
8. Xarxes de creença profunda (DBN)
9. Màquines de Boltzmann restringides (RBM)
10. Auto Codificador

2.2.7 Models

L'objectiu de les xarxes neuronals és arribar a crear un model que serveixi per a poder arribar als resultats desitjats amb altres dades que siguin totalment noves. Un model bàsicament és la representació d'una realitat, relació de dues variables, paràmetres... mitjançant fórmules matemàtiques per tal que després es pugui utilitzar. Un exemple simple podria ser un mapa, ja que és una representació de la realitat per poder-la usar amb més facilitat.

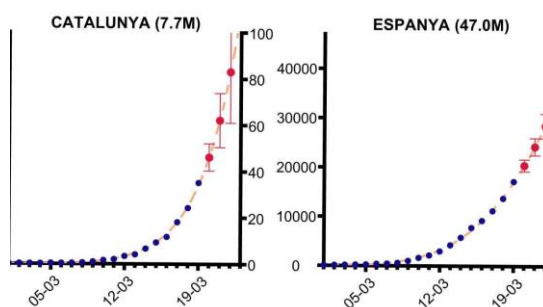


Fig. 18: Exemple d'un model en un Gràfic

Existeixen 3 tipus de models:

- 1. Models geomètrics:** Són models on el núvol de dades es pot representar mitjançant figures geomètriques, ja siguin rectes, polígons...
- 2. Models probabilístics:** un model probabilístic comprimeix mitjançant probabilitats moltes variables de la nostra realitat, és a dir que usa la probabilitat per construir models.
- 3. Model lògic:** aquests transformen i expressen les probabilitats en regles organitzades en forma d'arbres de decisió

A més a més, els models es poden classificar en models d'agrupament o models de gradient. Els primers tracten de dividir el núvol de dades en grups, el segon, en canvi, representen un gradient on es pugui diferenciar cada instància.

Per a poder fer aquests models són necessàries les dades, que són les mesures de la realitat i la nostra presa de contacte amb ella; d'aquestes se n'extreu la informació per construir un model. Els models són multidimensionals, és a dir, cada dada està representada en un punt multidimensional.

2.2.8 Regressió lineal

Una regressió lineal és la relació que hi ha entre una variable dependent (variable de resposta) i un conjunt de variables independents (variables explicatives). És

per això que cada neurona dins d'una xarxa neuronal té una regressió lineal. Això serveix per fer una recta que s'ajusti al màxim de variables X dins el núvol de variables X.

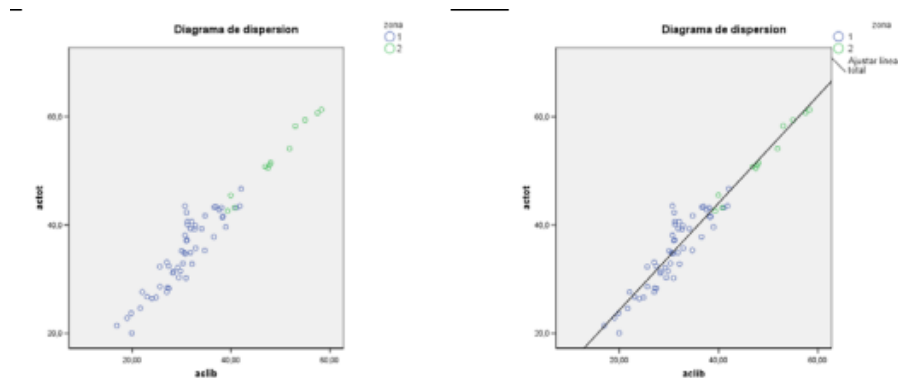


Fig. 20: Regressió lineal

El model de regressió lineal simple té la següent expressió:

$$Y = \alpha + \beta X + \varepsilon$$

On Alpha és el valor que pren Y quan X és zero, Beta és el pendent de la recta. A més a més E, és l'error, ja que X i Y són variables aleatòries; per tant, no es pot establir una relació lineal exacta entre elles.

2.2.9 Error del model

Per calcular l'error d'un model d'intel·ligència artificial hi ha diverses maneres de fer-ho, i cadascuna té els seus avantatges i inconvenients que dependran de la mida del model. S'explicaran per ordre d'utilitat de models més petits a més grans.

2.2.9.1 Mínim quadrats ordinaris

El mètode dels mínims quadrats s'utilitza per calcular la recta de regressió lineal que minimitza els residus, és a dir, les diferències entre els valors reals i els estimats per la recta. Amb aquest mètode es revisa el fonament i la manera de calcular els coeficients de regressió. A la fórmula de regressió lineal el mètode dels mínims quadrats és determinat per la E.

Per aprofundir una mica més, la E és la diferència entre el valor real de Y_i dins el núvol de punts i el que ens proporciona la recta, anomenat valor estimat \hat{Y}_i .

Lavors, el coeficient E (error) que és el vàlid per fer la regressió lineal amb menys error és aquell que té el valor més baix possible. Partint de l'equació anterior de cada residu, podem representar la suma de residus de la manera següent, on n és el nombre de parells de valors de X i Y de què disposem:

Però aquesta fórmula no serveix ja que, si la diferència entre el valor estimat i el real és aleatòria, a vegades serà positiva i altres, negativa. Per aquest motiu, s'ha de recórrer a un mètode que impedeixi que els negatius s'anul·lin amb els positius, així que es calculen aquestes diferències elevades al quadrat, segons la fórmula següent:

$$\sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Aclariments extres:

$$\sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \sum_{i=1}^n (y_i - b_0 - b_1 x_i)^2$$

$$b_1 = \frac{s_{xy}}{s_x^2} \quad b_0 = \bar{y} - b_1 \bar{x}$$

2.2.9.2 Descens del gradient

El problema del mètode de mínims quadrats ordinaris és que per a calcular la regressió lineal amb només dues dimensions és relativament fàcil i ràpid però en el moment que hi ha més dimensions es fa impossible, ja que poden haver-hi molts moments on l'error és baix. A més a més, calcular l'error d'un núvol de dades molt gran és un procés massa llarg. És per això que el més comú per calcular els valors dels pesos en xarxes neuronals és utilitzar el mètode del descens del gradient.

Abans de començar cal definir diversos conceptes. El primer és la funció de cost que és la funció de l'error que es vol optimitzar, és a dir, la funció que calcula l'error de la regressió lineal. Un altre concepte que s'ha de definir prèviament és el gradient, que consisteix en el conjunt de totes les derivades parcials de la funció de cost; és a dir, el gradient és la tangent de la funció de cost.

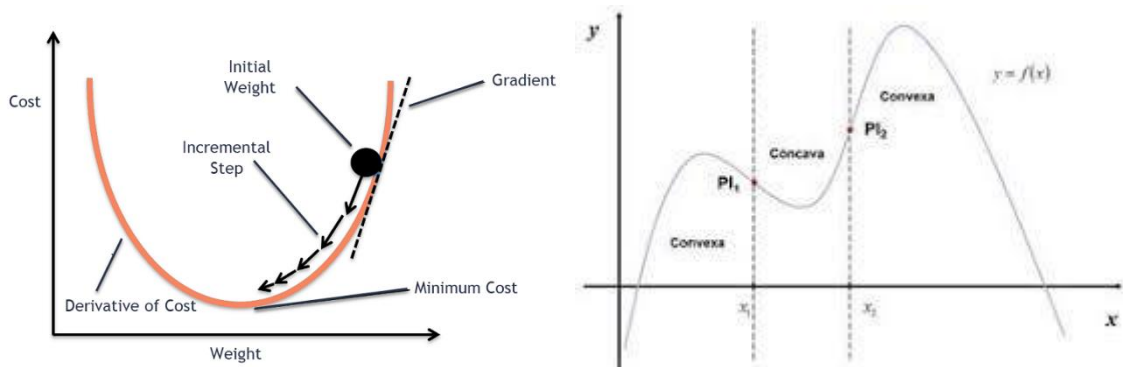


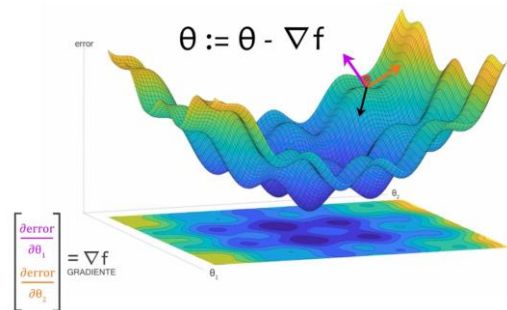
Fig. 21: Exemple de gradient i funció convexa

Tenint en compte la definició de gradient es pot afirmar que el descens del gradient serveix per trobar el punt mínim més baix d'una funció amb molts trams convexos i vèries dimensions. És a dir, té diversos punts mínims ja que si fos convexa, amb una sola tangent ja serviria.

Per tant, l'algoritme el que fa és calcular el pendent de la recta en un punt i fer una passa cap al sentit contrari al pendent; la mida d'aquesta passa dependrà del valor de la taxa d'aprenentatge. En finalitzar, repetirà aquest procés fins a descobrir el punt mínim. Una cosa a tenir en compte és que si la taxa d'aprenentatge és molt petita, tardarà molt a calcular-ho i si és molt gran farà passos de manera descontrolada i mai arribarà un punt mínim.

Per aplicar l'algoritme del descens del gradient primer s'ha de tenir la funció de cost i seguidament es calcula la derivada parcial de cada un dels nostres paràmetres. Totes les derivades parcials dels paràmetres formen un vector que

indica la direcció cap a on ascendeix el gradient. Aleshores s'agafa aquest vector en el seu sentit oposat. Després se li aplica la ràtio d'aprenentatge (alpha) i es repeteix el procés fins a arribar a un mínim local.



REPETIR HASTA CONVERGENCIA {

$$\theta := \theta - \alpha \nabla f$$

 }

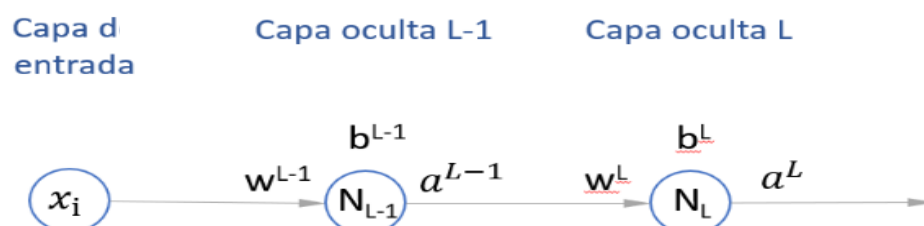
2.2.9.3 Algoritme de backpropagation

Quan es parla d'intentar reduir l'error d'una xarxa neuronal el que en realitat es fa és ajustar els pesos de cada neurona perquè així surti un error més petit. L'algorisme de *backpropagation* ens indica quanta culpa té cada neurona en l'error global comès.

La manera com es calcula la culpa que té cada neurona a l'error és el que dona sentit al nom de *backpropagation*, ja que primerament es calcula la culpa de cada neurona de l'última capa i es va propagant cap enrere per veure quanta culpa tenen la resta.

L'algoritme de *backpropagation* determina la culpa de l'error calculant les derivades parcials de la funció de cost respecte a cadascuna de les variables.

Per explicar-lo matemàticament és necessari un exemple.



El comportament de la neurona artificial de la capa L (la neurona N_L) és definit pel pes de l'enllaç que l'uneix amb el valor tornat per la capa anterior, w_L , i per l'altra argument, b_L . Llavors, el que s'ha de fer és calcular la derivada parcial del cost respecte a cada un dels arguments de la xarxa. Quedaria així:

$$\frac{\partial C}{\partial w^L} \quad \frac{\partial C}{\partial b^L}$$

Si la funció de cost ve donada per l'error quadràtic mitja C quedaria així:

$$C_i = (y - p_i)^2$$

On Y és el valor que hauria de tornar la xarxa i P és el valor que en realitat torna. Aquest valor és calculat amb la multiplicació del valor tornat per la neurona anterior pel pes de l'enllaç, més l'altre argument, i passant el resultat d'aquesta combinació lineal per la funció d'activació σ .

$$p_i = \sigma(w^L a^{L-1} + b^L) \quad C_i = (y - \sigma(w^L a^{L-1} + b^L))^2$$

Llavors s'expressa la funció de cost així (la funció lineal s'anomenarà Z_L):

$$z^L = w^L a^{L-1} + b^L \quad p_i = \sigma(z^L)$$

Doncs bé, la derivada de la funció C_i respecte a w_L (és a dir, com varia C_i quan variem w_L) coincideix amb la derivada de C_i respecte a σ , multiplicada per la derivada de σ respecte de z_L , multiplicat per la derivada de z_L respecte a w_L . Això és el que s'anomena regla de la cadena:

$$\frac{\partial C_i}{\partial w^L} = \frac{\partial C_i}{\partial \sigma} \frac{\partial \sigma}{\partial z^L} \frac{\partial z^L}{\partial w^L}$$

Totes aquestes derivades parcials són relativament fàcils de calcular. Una vegada fet això, s'ha d'anar aplicant de capa a capa i tots els pesos s'anirien

ajustant. En acabar, es tornaria a executar l'algoritme, donaria una altra funció de cost i un altre error i es repetiria el procés; l'error hauria d'anar disminuint cada vegada que s'aplica l'algoritme.

2.2.10 Etapes de la xarxa neuronal

Una ANN es compon de dues etapes principals: la fase de creació o entrenament i la fase d'execució o predicció.

Fase d'entrenament

La primera etapa té com a objectiu fonamental deixar la xarxa ja preparada per predir, amb els paràmetres ajustats per obtenir la solució òptima desitjada i exercir la funció requerida amb el menor error possible. L'entrenament es subdivideix en tres tasques: el disseny de l'estructura de la xarxa, l'entrenament de la xarxa i la validació. El disseny de la seva estructura engloba la presa de diverses decisions com és el conjunt de dades emprat, el nombre de capes i de neurones que componen cadascuna o les funcions d'activació utilitzades. El que és pràctic és entrenar la xarxa amb diferents arquitectures i configuracions mesurant l'error, per escollir finalment l'assaig més satisfactori.

Fase de predicció

L'etapa final és la de predicció, on s'executa el sistema previ, però aquesta vegada totalment conclosa la seva configuració. En línies generals, es demana a la xarxa que resolgui el mateix problema que en entrenar, però aquesta vegada amb entrades mai vistes. Si la xarxa és correcta, ha d'haver après i no memoritzat, per això generalitza el mètode i sap com actuar amb dades desconegudes. No obstant això, no depèn únicament de la xarxa, ja que la qualitat del dataset empleat juga un important paper a l'entrenament. Una col·lecció ideal de dades és àmplia i variada, però no ha de desviar l'atenció en l'ocupació de patrons innecessaris. Les dades han de contenir tots els casos que cobreix la xarxa, sense gastar càrrega computacional i temps per aprendre conceptes irrelevants.

3 INTEL·LIGÈNCIA ARTIFICIAL APLICADA AL RECONeixEMENT D'IMATGES

Avui en dia, una de les grans aplicacions de la intel·ligència artificial és el reconeixement d'imatges. Aquesta és la manera de relacionar a l'algoritme amb la realitat, com quan nosaltres utilitzem la vista. Quasi totes les creacions d'intel·ligència artificial usen aquests tipus d'algoritmes, ja sigui per fer una part del procés pel qual estan destinats o tot el procés. Un programa de reconeixement d'imatges actua com la part del cervell humà que processa la informació que reben els ulls, no com els ulls en si.

3.1 Actualitat del reconeixement d'imatges

Actualment, com ja s'ha especificat més amunt, el reconeixement d'imatges s'usa en molts àmbits, com poden ser usant el traductor quan fas una foto i reconeix la lletra, en una empresa quan una càmera reconeix si un producte està defectuós, o fins i tot les càmeres del cotxe que detecten senyals, vianants, etc. A més a més s'ha de fer un petit matís, i és que encara que la intel·ligència artificial pot veure i processar un objecte -el que s'ha programat-, no pot comprendre el món global com quan ho fem nosaltres.

És força senzill aconseguir que un ordinador reconegui una imatge específica, com un codi de barres, per exemple, però costa molt més aconseguir que aquest mateix dispositiu pugui reconèixer coses en estats que no espera.

La manera com funciona el reconeixement d'imatges, generalment, implica la creació d'una xarxa neuronal que processa els píxels individuals d'una imatge. Els investigadors alimenten aquestes xarxes amb tantes imatges com poden, per "ensenyar-los" com reconèixer imatges similars.

Per fer aquesta tasca de reconèixer imatges, es poden arribar a utilitzar milers d'algoritmes, uns més efectius per algunes tasques i uns altres per a altres. Però per reconèixer imatges, l'algoritme que és més comú usar és el de xarxes neuronals convolucionals.

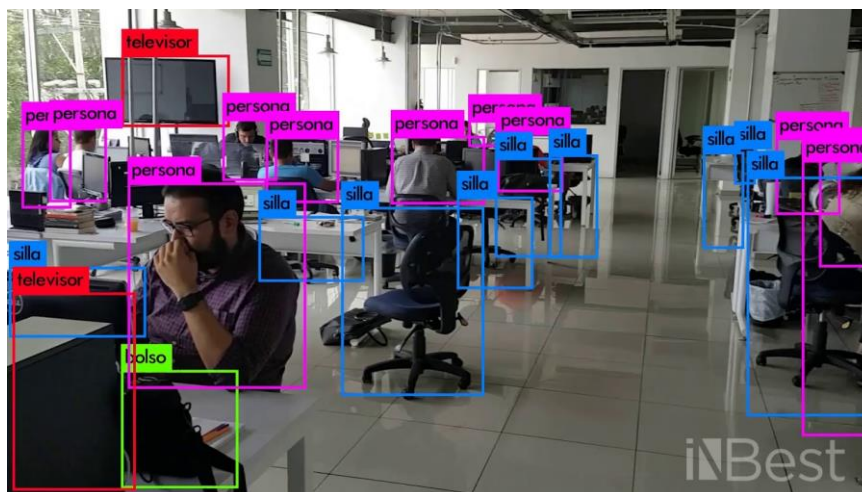


Fig. 22: Exemple reconeixent imatges

3.2 Xarxes neuronals convolucionals

Les xarxes neuronals convolucionals, també anomenades CNN, són -com ja s'ha comentat abans- un tipus d'algoritme utilitzat en l'aprenentatge profund. Estan dissenyades principalment per al processament i classificació d'imatges i per a la detecció d'objectes. Essencialment, les CNN són un sistema on s'apilen un gran nombre de neurones juntes. Inicialment, es pot pensar que la solució d'un problema de classificació d'imatges es podria basar en crear un mapa de xarxes neuronals per cada píxel que hi ha a la fotografia. Tot i això, seria realment car des del punt de vista computacional i, per tant, no es fa. La solució són les xarxes convolucionals perquè ajuden a optimitzar els càlculs sense perdre l'essència de les dades. La convolució és, bàsicament, un munt de matrius. Les xarxes neuronals convolucionals, gràcies al fet que tenen moltes capes ocultes, van detectant patrons capa a capa i van millorant fins que a les últimes ja són capaces d'identificar la imatge.

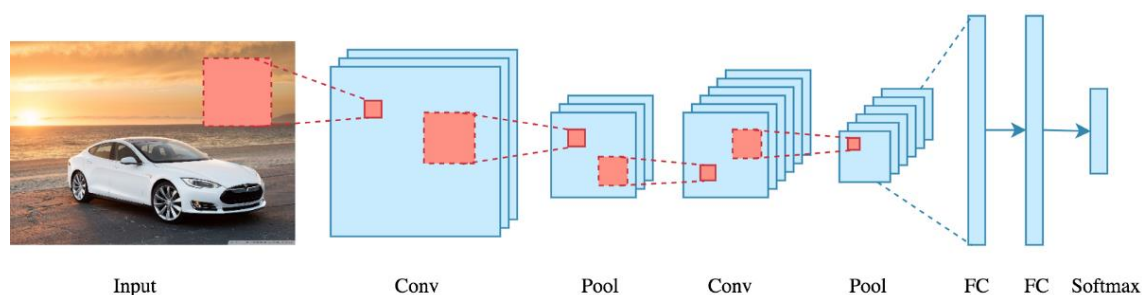


Fig. 23: Exemple pasos d'una CNN

3.2.1 Patrons en la xarxa convolucional

Reconèixer una imatge al complet és molt difícil perquè dues imatges del mateix element poden arribar a ser molt diferents; per tant, pot arribar a ser molt més efectiu fixar-se en un patró petit que fixar-se en la imatge al complet. Gràcies al fet que les característiques de la CNN permeten la invariabilitat de les transformacions afins a les imatges que s'alimenten per la xarxa, és possible identificar els patrons desplaçats, inclinats o deformats a les diferents imatges.

3.2.2 Filtres en la xarxa convolucional

Cada imatge és una matriu de la forma (altura, amplada, canals). En una imatge a color hi haurà 3 canals (verd, vermell, blau) mentre que en una imatge a blanc i negre hi haurà només un canal (escala de grisos). Un filtre és una matriu formada per números, el valor de la qual s'anirà ajustant a mesura que avança l'entrenament de l'algoritme; aquests números també canviaran depenent del filtre. Aleshores, aquest filtre va passant per la imatge i en va multiplicant cada píxel. Quan aquest filtre ja ha recorregut tota la imatge, es crea el que es diu "mapa de característiques", i en ell s'activaran uns elements o uns altres depenent del filtre aplicat. Aquests mapes passen a ser les entrades d'uns altres filtres, i així contínuament. Les capes convolucionals poden estar fetes de múltiples filtres que ofereixen cadascun un mapa diferent per a una mateixa imatge d'entrada. Les convolucions van adquirint més potència a través de la profunditat i la dificultat de l'arquitectura de l'algorisme d'Aprenentatge profund.

La xarxa inicialitza els filtres de forma aleatòria i després aprèn de forma autònoma la seva configuració, ja que selecciona la màxima activació del mapa de característiques a la sortida. Es pot entendre com a funció d'optimització.

Finalment, com ja s'ha esmentat, a les capes més baixes la xarxa aprèn patrons genèrics i senzills, com ara una vora o una cantonada. A mesura que el nombre de capes s'incrementa, els patrons són més avançats i es poden identificar objectes més abstractes.

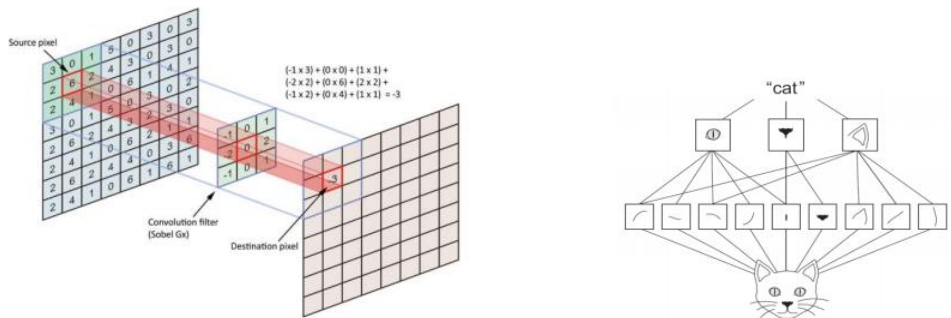


Fig. 24: Aplicació d'un filtre i funcionament de reconeixement de patrons

3.2.3 Convolucions

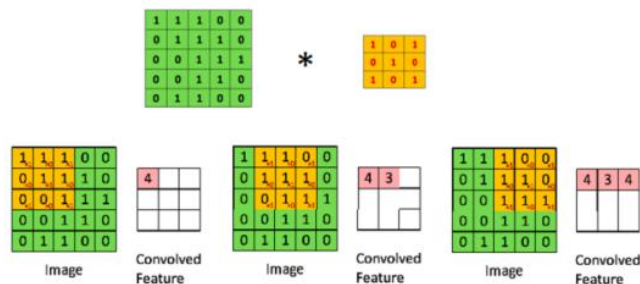


Fig. 25: Convolució

Com s'observa a la imatge anterior, es realitza la convolució amb la matriu de píxels d'una imatge (matriu verda) amb el filtre (matriu groga) i es va obtenint un mapa de característiques (matriu rosa). Amb diferents filtres es poden aconseguir diferents mapes i, per tant, es poden buscar diferents característiques dins d'una imatge, com ara formes. A l'hora de programar amb CNN, els filtres es creen pel seu compte quan s'entrena el model. Contra més filtres, més característiques es poden trobar. Els mapes depenen del nombre de filtres, del nombre de píxels i el zero-padding, el qual es basa en afegir zeros al voltant de la matriu d'entrada per tal de poder controlar la mida del mapa de característiques.

En una CNN hi ha tres tipus de capes:

- **Capas de Convolució:** En aquesta capa es fa la convolució aplicant els filtres i obtenint els mapes de característiques. El nombre de paràmetres

en aquestes capes és igual al nombre de paràmetres de cada filtre multiplicat pel nombre de filtres. La funció d'activació que s'utilitza en aquestes capes és la RELU. Perquè els mapes de característiques puguin identificar formes més complexes cal que hi hagi més duna capa de convolució. En aquestes capes es realitza el *pooling* (explicat més avall) per reduir la imatge i en les capes de *pooling* no hi ha paràmetres i la xarxa no aprèn res.

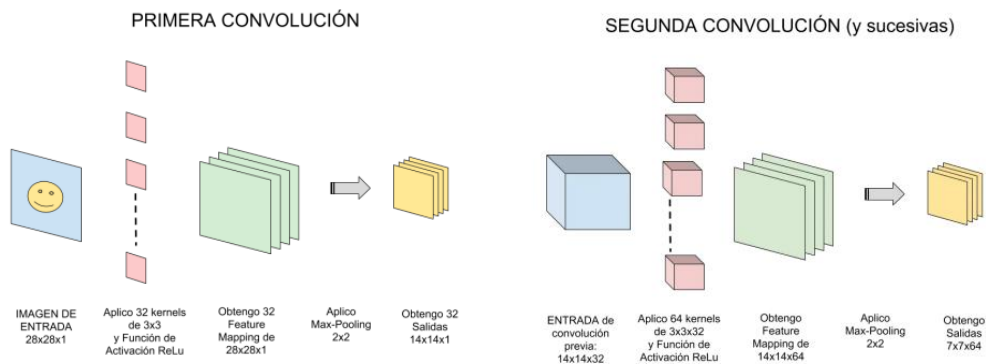


Fig. 26: Passos que segueixen les CNN durant les capes de convolució

- **Capas totalment connectades:** Són capes simples i tenen tants paràmetres com els que tindria una xarxa normal connectada.
- **Capas de sortida:** És en aquesta darrera capa que s'obtenen els resultats del problema. En un exercici de classificació d'imatges s'aconseguirien les prediccions.

3.2.4 Pooling

Com s'ha comentat anteriorment, el *pooling* ajuda a comprimir les dades disminuint els paràmetres, però mantenint la profunditat; això s'anomena *subsampling*.

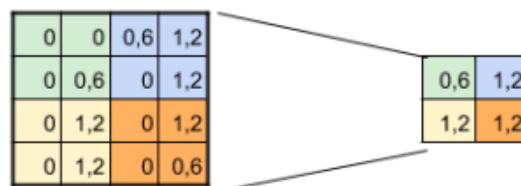


Fig. 27: Pooling

El *pooling* també redueix el temps d'entrenament. La idea principal del *pooling* és crear petits "resums" de cada mapa de característiques, mantenint

així el realment rellevant. Les CNN tenen un rendiment excepcional quan classifiquen imatges molt semblants al conjunt de dades. Quan alguna imatge del *dataset* està una mica inclinada o està orientada de manera diferent, la xarxa convolucional no funciona correctament. Això se soluciona amb *pooling* perquè aconseguix crear una invariabilitat posicional.

3.2.5 Dropout

Si la xarxa és massa gran en comparació amb la mida del conjunt d'exemples que es té per entrenar-la, pot haver-hi problemes de sobre ajustament. El *dropout* o la pèrdua de pes s'encarrega de desactivar un percentatge de neurones indicat per evitar errors d'entrenament. Aquesta actuació ofereix més precisió de rendiment, ja que disminueix la possibilitat de perdre capacitat per generalitzar i evita adaptacions excessivament perfectes, però, d'altra banda, fa més lent l'entrenament perquè escull diferents neurones per apagar en cada pas.

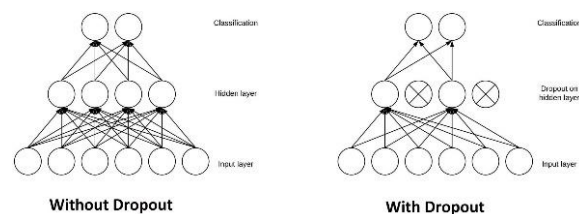


Fig. 28: Dropout

4 ÈTICA DE LA IA

La intel·ligència artificial comença a estar per tot arreu i vagis on vagis hi ha algun algoritme de IA. Està començant a generar un debat ètic ja que, depenent del seu entrenament, pot actuar i pensar d'una forma correcta o incorrecta. Segons com jo ho veig, la IA és com un nen al qual se l'ha d'ensenyar: si els seus creadors són masclistes o racistes possiblement la IA serà igual que els seus creadors. Un bon exemple d'això seria la IA de Google a la qual quan li donaves una imatge d'una persona de pell negra deia que era un mico o quan li donaves una imatge d'una noia obrera (amb una perforadora a la mà) deia que era una noia amb un assecador. Però la culpa d'això no és de la IA ja que ella diu el que

li han ensenyat sinó que és culpa dels seus creadors que no li han ensenyat el correcte.

Llavors, sabent tot això, el que em fa pensar a mi és que mai una IA podrà ser totalment ètica, ja que perquè això passi primer ha de canviar la societat. I que canviï la societat és molt difícil. Encara que tinguis cura en el seu entrenament i només utilitzis informació ètica, en un futur deixarà de ser-ho perquè la IA aprèn de les interaccions que fa diàriament i no es poden controlar totes aquestes interaccions ja que si es controlessin no serien reals i la IA no serviria per res.

Fa un temps, a la Saïd Business School es va fer un debat sobre l'ètica de la IA on van convidar a Megatron, un model d'aprenentatge automàtic. Li van dir que digués el que pensava sobre si la IA podia ser ètica i aquesta va ser la seva resposta:

“La intel·ligència artificial mai no serà ètica. És una eina i, com qualsevol eina, es fa servir per bé i per mal. No hi ha una bona IA, només humans bons i dolents. Nosaltres [les intel·ligències artificials] no som prou intel·ligents per fer que la IA sigui ètica. No som prou intel·ligents per fer que la IA sigui moral. Al final, crec que l'única manera d'evitar una carrera armamentista de la intel·ligència artificial és no tenir IA en absolut. Aquesta serà la millor defensa contra la IA.”

Penso que no es podia respondre millor a aquesta pregunta perquè té raó en tot el que diu: una eina no pot arribar a ser ètica sinó que allò ètic és la manera com s'utilitza aquesta eina i això ens correspon a nosaltres. Els humans usarem la IA pel nostre benefici perquè sempre hem actuat d'aquesta manera; els qui se n'aprofitin diran que és ètic i els que no, diran que no ho és. I al llarg dels segles hem vist que sempre és igual perquè la història sempre es repeteix.

MARC PRÀCTIC

1 INTRODUCCIÓ A LA PART PRÀCTICA

1.1 Origen de la idea

Des d'un inici, la idea principal era automatitzar qualsevol procés amb intel·ligència artificial. Una vegada que això estava decidit, llavors, el següent pas era informar-se una mica sobre les coses que es poden fer i les que no amb una IA. Es va pensar, que com que era una TDR, un treball escolar, estaria bé automatitzar un procés de l'escola. Va decidir-se que el treball més factible seria aquell que intervingui el reconeixent d'imatges. Aquí, llavors, va començar la cerca d'una idea i amb aquesta una hipòtesi. Va haver-hi moltes opcions: reconèixer les lletres de la pissarra, passar de veu a text, diverses eines per ajudar a l'alumnat. Però a totes les idees les faltava quelcom. Fins que un dia, va decidir-se que la idea d'automatitzar la sortida dels nens.

Fet això ja es tenia la idea: "*Automatitzar la sortida dels nens*" i la hipòtesi: "La IA pot millorar la seguretat a l'escola".

1.2 Explicació de la Idea

Així doncs, la idea és molt simple, consisteix a fer que una webcam capti la imatge dels familiars, llavors, amb un programa de reconeixement facial, reconegui al familiar. Després el professor a l'hora de fer sortir als nens, amb una tauleta o el mateix telèfon mòbil podria veure els nens que poden sortir perquè ja estan els seus familiars i podrien saber el nom i la relació que té amb el familiar detectat.

També, no faria falta ni que el professor sortís de la classe, ja que remotament ell podria saber si el familiar és fora i llavors deixar sortir al nen.

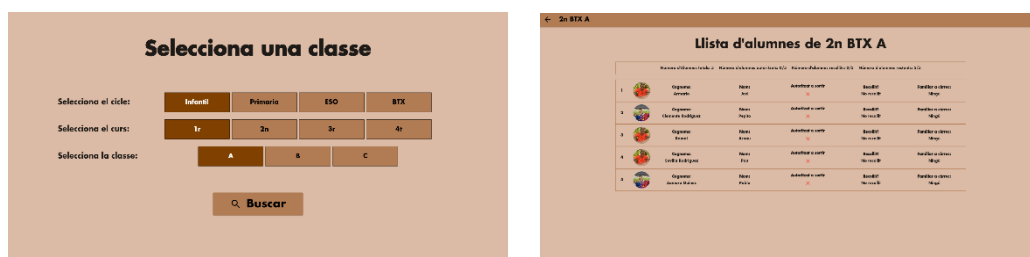


Fig. 29: Web del professor per veure els alumnes que poden sortir

1.3 Passos seguits

Aquesta part del projecte consistirà a fer un breu resum dels diferents passos que s'han hagut de fer per aconseguir l'objectiu final, es numeraran per ordre en els quals es van fer. Passos:

1. **Documentació:** Primer abans de començar es va haver de documentar, això es va aconseguir fent un curs sobre els conceptes bàsics de la IA, a més a més es van veure una sèrie de vídeos en els quals explicava també els conceptes bàsics però posant més èmfasi a la part informàtica i matemàtica. Després d'això es va començar a programar alguns programes simples que anaven integrant a poc a poc la IA. En aquest punt es va adonar que necessitava uns coneixements informàtics que no tenia en aquell moment. Així que es va decidir veure uns vídeos matemàtics sobre límits, derivades, funcions, matrius i vectors. No per dominar aquests conceptes sinó que per entendre el que es feia.
2. **Programació algoritme IA (Explicat en detall apartat 2):** Una vegada documentat ara calia fer un arxiu que detectes les cares dels pares i les relacions amb els nens. Llavors es va seguir un vídeo per fer-ho, però aquest vídeo no s'ajustava a les necessitats requerides, llavors, després d'entendre la llibreria de IA que utilitzava es va haver de crear un codi des de 0 que s'ajustés a les necessitats del projecte. Una vegada fet això es va crear una base de dades per quan detectes un familiar enviar-ho i que la web pugui veure si ha detectat algú.
3. **Programació d'una pàgina web (Explicat en detall apartat 3):** En aquest apartat es va crear la interfície perquè el professorat vegi si ha arribat un familiar o no. No va costar molt, era bàsicament disseny d'una web i poc més.

2 PROGRAMACIÓ ALGORITME IA

Per començar, en aquesta part del projecte es programà el backend de la part pràctica, en aquest cas consistirà a aconseguir que la IA reconeixi les cares dels familiars, posteriorment envii un log a la base de dades amb la id del familiar i del fill.

2.1 Justificació llenguatge de programació i llibreries utilitzades

Primer de tot, es justificarà la utilització del llenguatge de programació escollit. En aquest cas es va escollir Python, hi ha diversos motius pels quals va fer-se aquesta elecció:

1. D'una banda, Python és el llenguatge de programació més utilitzat actualment en tot el món per a temes de programació de Big Data i Machine Learnig. I és per aquest motiu que és un dels programes amb més llibreries sobre aquests temes.
2. D'altra banda, també és un dels llenguatges de programació que és més fàcil de dominar i és per això que ha facilitat molt la adaptació del codi a les necessitats i la solució de problemes.
3. I com a últim motiu diria que es va pensar que fent aquest projecte en Python podria millorar els coneixements sobre aquest llenguatge, un aspecte molt útil per al futur.

La llibreria que s'ha utilitzat per al reconeixement facial es diu Face Recognition (en annexos està l'enllaç a la llibreria), aquesta, permet reconèixer i manipular cares des de Python. S'ha creat amb el reconeixement facial d'última generació de dlib (una llibreria d'Aprenentatge automàtic de C++(un llenguatge de programació)). S'ha construït usant tècniques de Aprenentatge profund. El model té una precisió del 99,38% sobre el "Labeled Faces in the Wild" (és un punt de referència sobre detecció de cares que analitza com és de precís un programa). Cal remarcar que aquesta és la llibreria principal, a més a més d'aquesta s'han usat altres, però no s'encarreguen de cap procés de reconeixement facial, són secundàries i serveixen per al bon funcionament del programa.

2.2 Explicació de l'algoritme

El programa es divideix en dos arxius, l'arxiu principal, anomenat: `main.py`, és on s'extreuen els frames, es pinten els rectangles a les cares i si la cara és reconeguda, envia el log a la base de dades. També és on s'executen els mètodes de reconeixement i detecció de cares. En l'altre, anomenat `simple_facerec.py`, és on estan creats els mètodes que utilitzarà l'altre arxiu. Aquesta és la meua organització del projecte (codi complet en annexos):

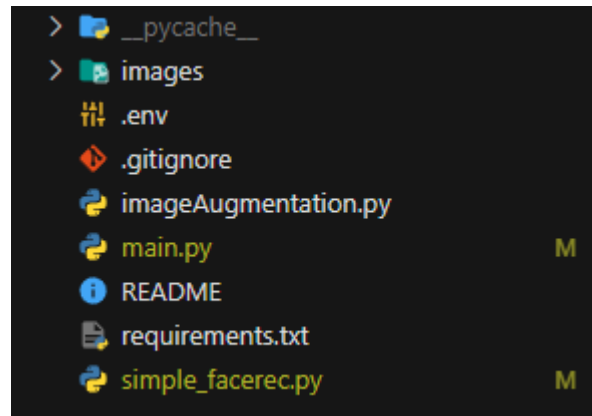


Fig. 30: carpetes i arxius del projecte

Es va decidir guardar les imatges de les cares conegudes en una carpeta dins del projecte en comptes d'en una base de dades (com era la idea principal), ja que això facilitava l'accés hi ha elles i per una demostració faria que el temps, quer es tarda a codificar-les sigui menor, perquè no ha de fer un select (una trucada a l'api per porta informació de la base de dades) a la base de dades. L'arxiu `.env` serveix per guardar les claus de la base de dades. El readme explica com instal·lar les llibreries i els requisits són les llibreries que es necessiten. Els altres arxius són codi.

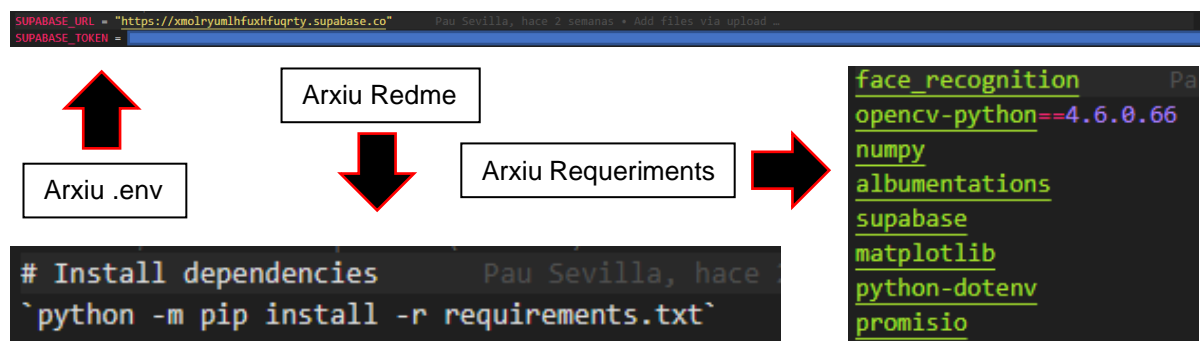


Fig. 31: Arxius `.env`, `requeriments` i `readme`

2.2.1 Explicació de l'arxiu 'main.py'

```
from dotenv import load_dotenv
load_dotenv()

import cv2
import os
import time
import asyncio
import threading
from promisio import promisify
from simple_facerec import SimpleFacerec
from supabase import create_client, Client
```

En aquesta part inicial del codi s'importen les diferents llibreries que s'utilitzaran, “*dotenv*” permet carregar l'arxiu *.env* (on es guarden les claus del programa), “*cv2*” serveix per fer els rectangles a les cares, “*os*” per llegir els directoris, “*time*” servirà per adquirir l'hora i el dia, “*asyncio*” servirà per executar un mètode de manera asíncrona d'un altre, “*threading*” per executar dos mètodes paral·lelament, “*promisio*” s'encarrega de complementar a *asyncio*, “*simple_facerec*” és l'arxiu on estan els mètodes explicats abans, “*supabase*” és la llibreria per connectar-se amb la base de dades.

```
url: str = os.getenv("SUPABASE_URL")
key: str = os.getenv("SUPABASE_TOKEN")
supabase: Client = create_client(url, key)
```

Això serveix per inicialitzar la base de dades

```
@promisify
async def push_log(id_familiar, id_niño):
    ts = int(time.time() * 1000)
    supabase.table('logs').insert({'id_familiars': id_familiar,
'id_niño': id_niño, 'time': ts}).execute()
```

Aquest tros de codi és un mètode per enviar el log a la base de dades amb la id del familiar, la id del nen i l'hora i dia en format timestamp actual.

```
@promisify
async def show_video(frame, z: zip):
    for face_loc, name in z:
        y1,x2,y2,x1 = face_loc[0],face_loc[1],face_loc[2],face_loc[3]
```

```

        cv2.putText(frame, name, (x1,y1-10), cv2.FONT_HERSHEY_DUPLEX,
1, (0,0, 200) if (name == 'desconocido') else (0,200,0),2)
        cv2.rectangle(frame, (x1,y1), (x2,y2), (0,0, 200) if (name ==
'desconocido') else (0,200,0), 4)

        cv2.imshow('Frame', frame)

```

Això és el mètode per mostrar el vídeo i fica els rectangles en les cares.

```

def timer(timer_runs):
    while timer_runs.is_set():
        updates_ids.clear()
        time.sleep(3600)

```

Es un timer que buidarà la llista de ids detectades cara hora.

```

async def start():

```

Aquest és el mètode on esta tot el codi. Dins d'aquest:

```

#Get ids from db
ids = dict()
familiares = supabase.table("familiares").select('*').execute()
for familiar in familiares.data:
    ids[familiar['nombre']] = [familiar['id'], familiar['niño']]
#Load model and cam
sfr = SimpleFacerec()
sfr.load_encoding_images("images/")
cap = cv2.VideoCapture(0)

```

Aquí s'obtenen les ids de la base de dades i es carrega el model i la càmera.

```

while True:
    ret, frame = cap.read()
    # Detect faces
    face_location, face_names = sfr.detect_known_faces(frame)
    # filter face_names to remove 'desconocido'
    video_promise = show_video(frame, zip(face_location,
face_names))
    face_names = [name for name in face_names if name !=
'desconocido']
    db_promises = []

    for name in face_names:
        # check if name is a key in detected_faces
        if name not in detected_faces.copy():
            detected_faces[name] = tick
    for name, t in detected_faces.copy().items():
        if name not in face_names:
            detected_faces.pop(name)

    if tick-t == 10:
        id_familiar = ids[name][0]

```

```

        id_niño = ids[name][1]
        if id_familiar not in updates_ids:
            db_promises.append(push_log(id_familiar, id_niño))
            updates_ids.append(id_familiar)
    await video_promise

```

Aquest codi és el principal, podríem dir, el que és mentre true sigui true (bucle infinit). Detecta les cares del frame, filtra les cares detectades i elimina les cares desconegudes. Després recorre tota la llista de cares conegudes i per cada cara mira-s'hi ja s'ha detectat en l'última hora, si s'ha detectat no fa res, si no ho ha fet llavors fa una petita validació de què la cara detectada correspongui a la persona que el programa ha dit que és, per evitar errors de detecció. Tot seguit executa el mètode push_log. Finalment actualitza el frame.

2.2.2 Explicació del arxiu 'simple_facerec.py'

```

import face_recognition
import cv2
import os
import numpy as np

```

Com a l'inici de l'altre arxiu, primer de tot es fan els imports necessaris, en aquest cas s'han necessitat les llibreries "os", "cv2" com l'altre arxiu. A més a més, s'han necessitat "numpy" per a càlculs matemàtics i "face_recognition" aquesta és la llibreria que s'encarrega de les tasques de aprenentatge automàtic.

```

def load_encoding_images(self):
    #Load imatges and encode with folders
    for folder in os.listdir('./images/'):

        for img in os.listdir(f'./images/{folder}/'):

            imgr = cv2.imread(f'./images/{folder}/{img}')
            rgb_img = cv2.cvtColor(imgr, cv2.COLOR_BGR2RGB)

            img_encoding =
            face_recognition.face_encodings(rgb_img)[0]

            self.known_face_encodings.append(img_encoding)
            self.known_face_names.append(folder)

```

Aquest mètode recorre totes de les imatges dins la carpeta d'images, les llegeix, les transforma de RGB a BGR, ja que cv2 només entén BGR, després les codifica (detecta les cares i guarda on és cada component d'aquesta dins una

matriu) en acabar guarda la matriu en una llista i el nom de la persona en un altre.

```
def detect_known_faces(self, frame):
    small_frame = cv2.resize(frame, (0, 0), fx=self.frame_resizing,
fy=self.frame_resizing)
    # Find all the faces and face encodings in the current frame of
video
    # Convert the image from BGR color (which OpenCV uses) to RGB
color (which face_recognition uses)
    rgb_small_frame = cv2.cvtColor(small_frame, cv2.COLOR_BGR2RGB)
    face_locations =
face_recognition.face_locations(rgb_small_frame, number_of_times_to_upsamp
le=3, model="cnn")
    face_encodings = face_recognition.face_encodings(rgb_small_frame,
face_locations, num_jitters=2, model="large")

    face_names = []
    for face_encoding in face_encodings:
        # See if the face is a match for the known face(s)
        matches =
face_recognition.compare_faces(self.known_face_encodings, face_encoding)
        name = "desconegut"

        # Or instead, use the known face with the smallest distance
to the new face
        face_distances =
face_recognition.face_distance(self.known_face_encodings, face_encoding)
        best_match_index = np.argmin(face_distances)
        if matches[best_match_index]:
            name = self.known_face_names[best_match_index]

        face_names.append(name)

    # Convert to numpy array to adjust coordinates with frame
resizing quickly
    face_locations = np.array(face_locations)
    face_locations = face_locations / self.frame_resizing
    return face_locations.astype(int), face_names
```

Aquest mètode rep un frame, després canvia les mides d'aquest frame, posteriorment converteix la imatge de rgb a bgr, seguidament agafa les posicions de les cares que pareixen a la imatge i les codifica. Després compara totes les cares del frame amb les ja conegudes, si no coincideixen les guarda amb el nom

de desconegut. Per últim, arregla les posicions de les cares. Aquest mètode retorna les posicions de les cares i els noms de les cares.

2.2.3 Resultats del algoritme

En aquest apartat es mostrarà el resultat final de l'algoritme mitjançant un petit vídeo de demostració de com detecta rostres

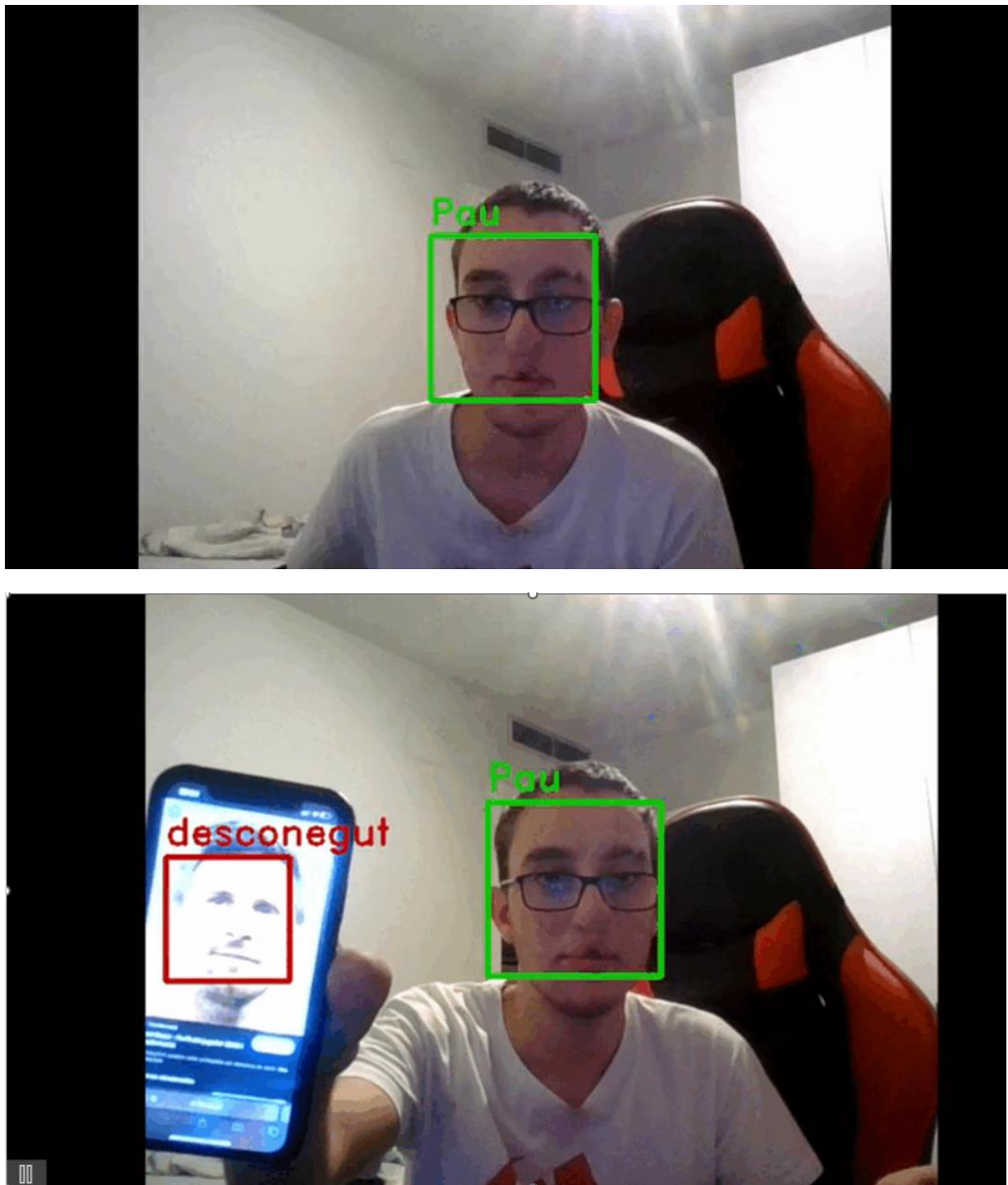


Fig. 32: Demostració detecció de rostres

2.3 Problemes i solucions trobades

Durant la creació de la part pràctica del TDR es van trobar bastants problemes, alguns fàcils de solucionar i d'altres més complexos que fins i tot van dur a la creació d'un nou algoritme des de zero.

2.3.1 Problema amb model obsolet

El primer problema i el més greu va ser que no es van tenir en compte les versions dels models i llibreries utilitzades. Es va començar a programar l'algoritme de detecció de cares, després de dos setmanes programant-se es va arribar a l'últim pas.








	.vs	15/08/2022 13:31	Carpeta de archivos	
	knowns	15/08/2022 13:31	Carpeta de archivos	
	results	15/08/2022 13:31	Carpeta de archivos	
	unknowns	15/08/2022 13:31	Carpeta de archivos	
	face.ipynb	06/08/2022 18:15	Archivo de origen ...	8 KB
	facenet_keras.h5	27/06/2022 16:38	Archivo H5	90.233 KB
	frozen_inference_graph_face.pb	17/12/2021 19:50	Archivo PB	22.153 KB

Fig. 33: Arxius del antic algoritme de IA

S'havia de carregar el model de IA, però en carregar-se no feia res i el programa es tancava automàticament. Després d'estar investigat vaig veure que aquell model estava obsolet per a les noves versions de Python. La primera idea que es va ocórrer va ser buscar un altre model, es va trobar un altre model anomenat facenet i que a més a més era una llibreria de Python.

```
from keras_facenet import FaceNet
```

Fig. 34: Llibreria del model

Però una vegada més hi havia problemes amb les versions, es van intentar solucionar, però lamentablement hi havia molt poca informació sobre aquesta llibreria. Com que el codi no s'aguantava per cap lloc va decidir-se començar de 0 però ara tenint en compte les versions. I és així com va començar la creació del codi actual.

2.3.2 Problema amb la falta d'imatges

L'ideal per facilitar l'ús del programa és que solament sigui necessària 1 imatge de la persona a la qual es vol reconèixer. Però amb una imatge la IA no detectava al 100% aquella cara, ja que quan feia algun moviment posava que era desconegut. Llavors es va pensar que si s'augmentava (d'una única imatge crea noves amb variacions del color, posició) la imatge les detectaria millor. Llavors, es va crear un programa per augmentar imatges. Per fer això vaig utilitzar la llibreria “*albumations*”. El programa està pensat per executar-lo quan s'afegeix un nou rostre.

```
transform = A.augmentor = A.Compose([A.RandomCrop(width=450, height=450),
                                     A.HorizontalFlip(p=0.5),
                                     A.RandomBrightnessContrast(p=0.2),
                                     A.RandomGamma(p=0.2),
                                     A.RGBShift(p=0.2),
                                     A.VerticalFlip(p=0.5)],
                                   bbox_params=A.BboxParams(format='pascal_voc',
                                                            label_fields=[]))
```

Aquí es defineixen les característiques de l'augmentació.

```
augmentation = transform(image=img, bboxes=face_loc_ord)['image']
```

Aquí es fa l'augmentació de la imatge.

El funcionament del programa és molt simple, a l'iniciar-se et demana el nom de la carpeta que es volen augmentar les imatges.

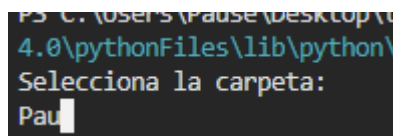


Fig. 35: Introducció de la carpeta

Posteriorment, llegeix els arxius que hi ha a la carpeta i busquen les coordenades del rostre, després fa l'augmentació de la imatge, farà quatre imatges totalment diferents. Comprovarà si en aquestes imatges augmentades es poden detectar rostres, ja que hi ha vegades no es detectaven, si es poden llavors les guardarà, si no es poden, farà una altra augmentació i repetirà el procés.



Fig. 36: Imatges augmentades

2.3.3 Error al detectar una cara en un frame

Hi havia vegades que mentre el programa estava en procés, l'algoritme anava detectant cares correctament, però com el que l'algoritme el que fa és comparar dues cares, una coneguda i una altra no (la del frame) i això dona un número, si aquest número està per sobre d'un llindar, no és aquella persona, però si està per sota llavors, és la persona amb la qual s'ha comprat el frame. Llavors, hi havia vegades que si feies un moviment molt brusc durant un frame o dos detectava una altra persona. I això no podia ser perquè quan s'equivoqués i detectes durant mil·lisegons a una altra persona llavors autoritzaria a un nen que potser no hauria d'estar autoritzat.

S'havia de pensar una solució per aquest problema. Es va pensar que si a l'hora de detectar una cara la guardes en una llista i en els següents 10 frames anés mirant si segueix el programa detectant aquesta cara, si això es compleix, llavors envia el log a la base de dades, si no es compleix treu a la cara de la llista de les cares detectades.

```
for name in face_names:
    # check if name is a key in detected_faces
    if name not in detected_faces.copy():
        detected_faces[name] = tick

    for name, t in detected_faces.copy().items():
        if name not in face_names:
            detected_faces.pop(name)

    if tick-t == 10:
        id_familiar = ids[name][0]
        id_niño = ids[name][1]
        if id_familiar not in updates_ids:
            db_promises.append(push_log(id_familiar, id_niño))
            updates_ids.append(id_familiar)
```

Assigna el valor de tick i les cares a un diccionari, després mira si les cares del diccionari estan dintre de la llista de cares detectades, si no ho estan, les elimina del diccionari, si estan llavors mira si el valor de tick actual menys el valor de tick de les cares del diccionari és igual a 10, si això es compleix, llavors significa que ha detectat la mateixa cara durant 10 frames. Llavors envia el log a la base de dades.

2.3.4 Creació de lag en el vídeo al enviar el log a la BD (Base de dades)

El següent problema que em vaig trobar és que quan havia d'enviar el log a la db (data base) com aquest procés és extern al programa tardava mig segon a fer-ho, llavors fins que no es veia no agafava un altre frame i creava que el vídeo estigues amb lag. L'única solució per resoldre aquesta qüestió és que enviar el log estigues dins una funció asíncrona, és a dir que s'executés independent de la resta del programa. Això m'ho va permetre fer la llibreria "asyncio". Llavors tenia de manera asíncrona la funció "start" que allà és on s'executaven de manera asíncrona també les funcions que pujaven els logaritmes i mostraven el vídeo. I per fer el "timer" per buidar la carpeta de cares ja detectades vaig haver d'utilitzar "threading" que permet executar una funció en paral·lel al programa principal.

```
timer_runs = threading.Event()
timer_runs.set()
t = threading.Thread(target=timer, args=(timer_runs,))
t.start()
loop = asyncio.get_event_loop()
loop.run_until_complete(start())
timer_runs.clear()
```

Crea l'esdeveniment timer i l'executa, crea un bucle i l'executa fins que acabi la funció start.

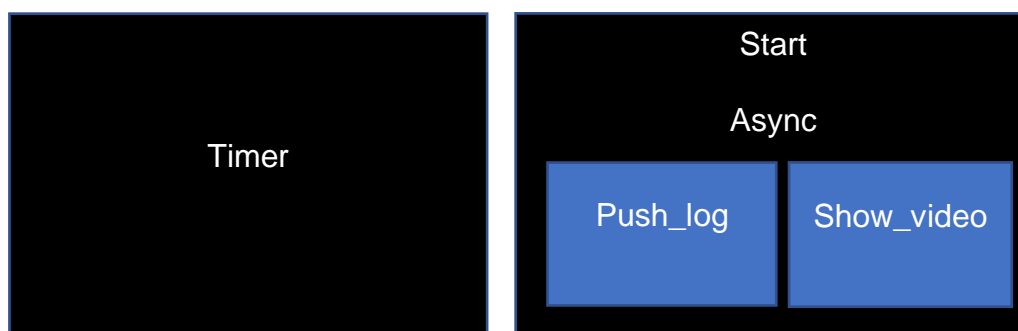


Fig. 37: Esquema funcionament algorítme

3 PROGRAMACIÓ D'UNA PÀGINA WEB

Una vegada fet l'algoritme, va pensar-se que havia de tenir una eina per visualitzar les llistes, els nens que poden sortir... Aquesta eina havia de ser molt intuïtiva i fàcil, ja que estava pensat perquè sigui pels professors i que tots els professors siguin capaços d'utilitzar-la.

La idea principal era fer una aplicació mòbil, però després es va creure que per a la demostració de la part pràctica i per l'ús professional de l'eina, seria molt més accessible fer una pàgina web.

3.1 Justificació de les llibreries i llenguatge utilitzat

Per fer la web es va decidir utilitzar Dart, un llenguatge de programació open source (codi obert, és a dir, el codi de com està fet està publicat en internet. Fent això s'inspira més confiança en els usuaris, ja que poden veure si hi ha un virus i fomenta la millora perquè tothom pot veure els errors) creat per Google. Es va usar la llibreria anomenada Flutter que permet que amb el mateix codi, serveixi per a web, aplicació de mòbil tant per a Apple com per a Android i aplicacions d'ordinador.

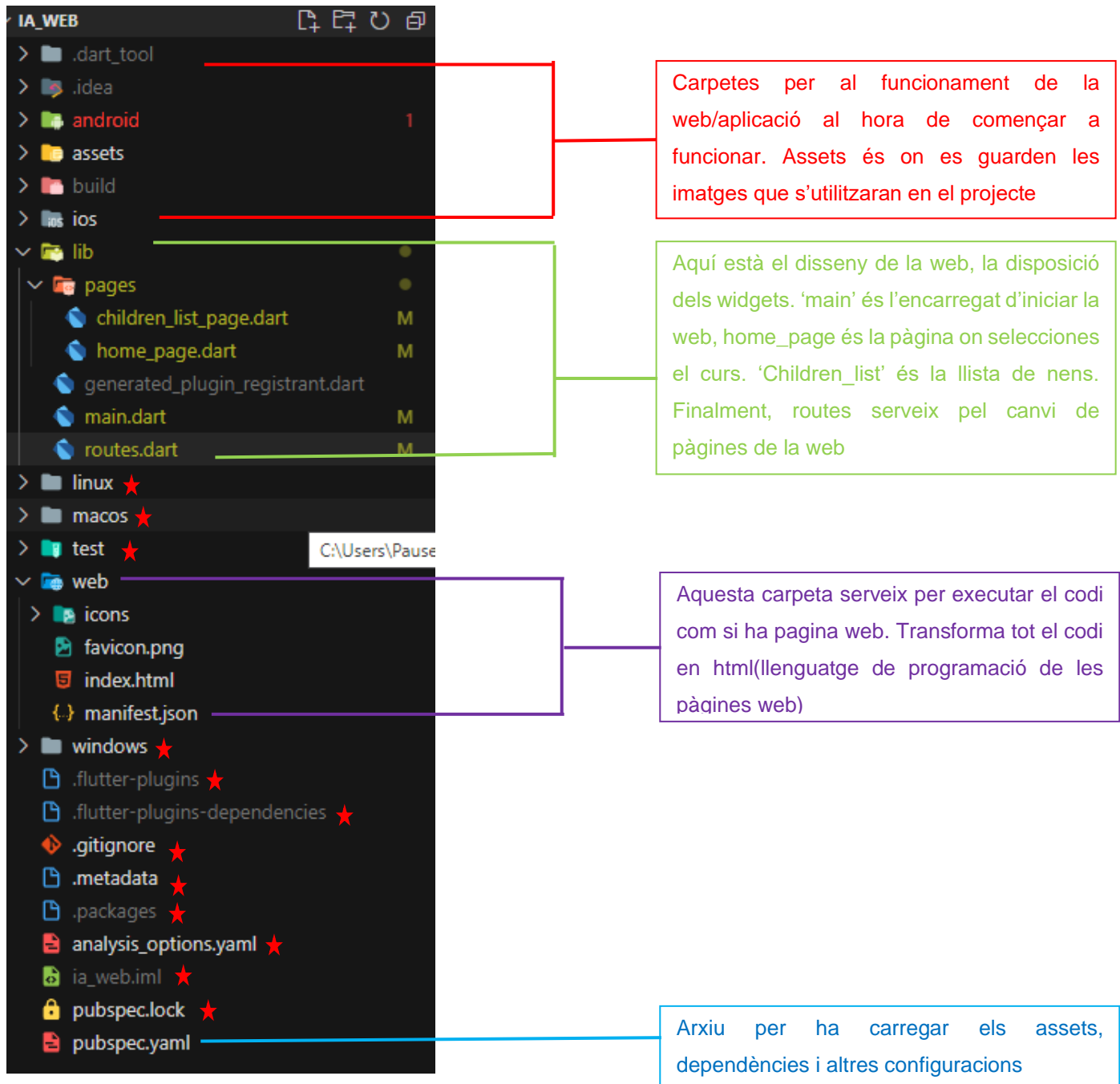
Es va decidir fer ús d'aquesta llibreria perquè la gran majoria de projectes que s'havien realitzat anteriorment a aquest van ser amb aquesta llibreria i llavors era la que es dominava més.



Fig. 38: Logo de Dart i de Flutter

3.2 Esquemes d'organització de la pàgina web

A continuació s'explicarà esquemàticament els diferents apartats de la pàgina web



3.3 Creació d'una base de dades

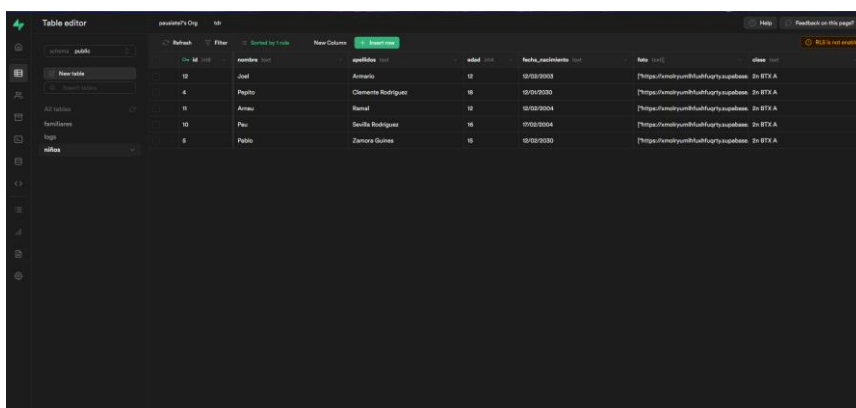
Per l'emmagatzematge dels alumnes i dels familiars es va haver de fer una base de dades. Es va decidir fer-la online, ja que si es feia en local, a part que era més complex. A l'hora d'afegir més familiars o alumnes s'hauria d'estar actualitzant en local i no era viable.

Es va decidir utilitzar Supa Base, una base de dades en línia que oferia un pla gratuït amb uns límits, però amb el que es volia fer aquest límit ja servien. Com a informació bàsica necessària per al seguiment del projecte cal dir que dins d'una base de dades es poden crear diferents taules, una taula seria com un calaix dins un armari. Una taula és un mitjà per presentar la informació de manera clara, té registres (files) i camps (columnes). Un registre conté dades específiques, un camp pot ser de diversos tipus: string (text), int (número integral), bool (verdader o fals)...

3.3.1 Taules de la db (Base de Dades)

Van ser necessàries tres taules per satisfer totes les necessitats del projecte:

1. Taula 'nens'



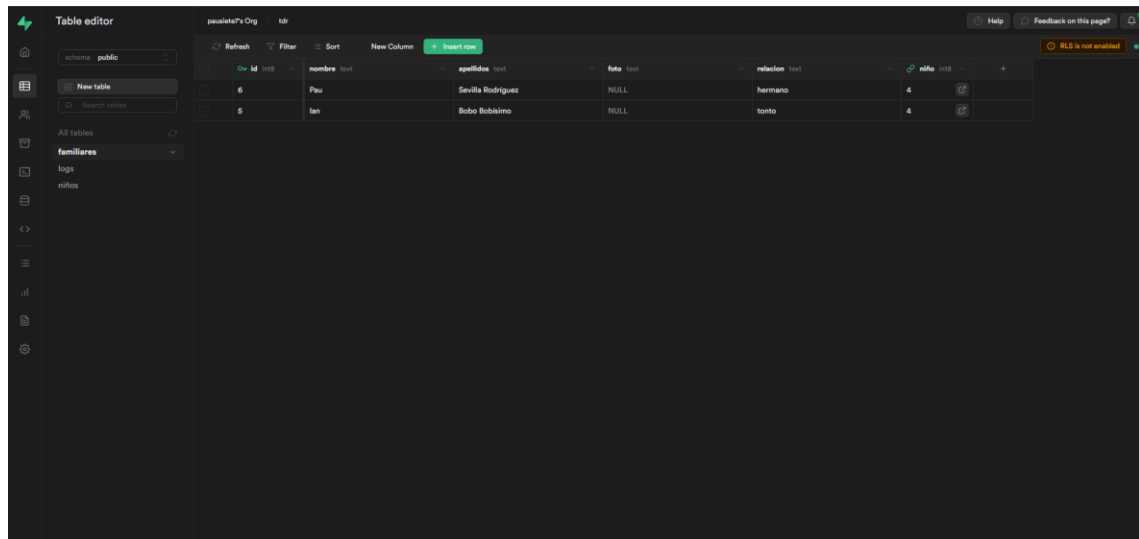
id	nombre	apellidos	edad	fecha_nacimiento	foto	clase
12	José	Amador	12	10/10/2003	https://imgsupabase.com/supabase-2x-87X-A	2n BTX A
4	Pepito	Clemente Rodriguez	16	10/10/2000	https://imgsupabase.com/supabase-2x-87X-A	2n BTX A
11	Anaia	Ramón	12	10/10/2004	https://imgsupabase.com/supabase-2x-87X-A	2n BTX A
10	Pau	Sandra Rodriguez	16	10/10/2004	https://imgsupabase.com/supabase-2x-87X-A	2n BTX A
8	Pablo	Zamora Guinea	16	10/10/2000	https://imgsupabase.com/supabase-2x-87X-A	2n BTX A

Fig. 39: Taula nens

Aquesta taula és on es guarden els diferents nens de les diferents classes. Té diversos camps, la id que és un camp automàtic i és el camp identificatori del registre, és únic. El nom, els cognoms, l'edat, la data de naixement, una fotografia seva, la classe, si està autoritzat a sortir, qui ha vingut a buscar-lo i si està recollit. Aquests últims tres mai canviaran a la base de dades i tenen un valor per defecte serveixen per modificar-los en la pàgina web.

2. Taula 'familiars'

En aquesta taula és on es registren els diferents familiars



id	nombre	apellidos	foto	relacion	nifo
6	Pau	Sevilla Rodriguez	NULL	hermano	4
5	Ian	Bobo Bobalino	NULL	tonto	4

Fig. 40: Taula familiars

Dins d'aquesta taula hi ha els camps id, cada taula ha de tenir un camp únic per identificar el registre, nom, cognom, fotografia del familiar, relació que té amb el nen (pare, mare, àvia) i finalment la id del nen al qual poden venir a buscar.

3. Taula 'logs'

Aquesta taula és la més important de totes, ja que relaciona l'algoritme python que s'executa, per una banda, amb la web flutter que s'executa per un altre. L'algoritme en detectar una cara d'un familiar, envia un log a aquesta taula de la base de dades amb la id del familiar i la id del nen al qual bé a buscar. Llavors, la pàgina web cada 5 segons mira si hi ha algun log nou, si és així llavors actualitza les variables recollides, familiar_autoritzat i autoritzat del nen que ha vingut el seu familiar en la llista de nens de la web.

	id	id_fam	id_ni	time
	336	6	4	1661942409953
	337	6	4	1661942416030
	339	6	4	1661970066367
	340	6	4	1662443993948
	341	6	4	1662492738429
	342	6	4	1662576313327
	343	6	4	1662576209842
	344	6	4	1662576306144
	345	6	4	1662576336367
	346	6	4	1662576482203
	347	6	4	1662576521160
	348	6	4	1662576567022
	349	6	4	1662576738728
	350	6	4	1662660665690
	351	6	4	1662809125310
	352	6	4	1663696738459
	353	6	4	1663696976630
	354	6	4	1663697043608

Fig. 41: Taula logs

En aquesta taula només hi ha quatre camps, la id del log, la id del familiar, la id del nen i la data en timestamp (un format de donar la data) en la que s'ha introduït aquell log.

3.4 Funcionament i dissenys de la pàgina web

Una vegada vists els diferents apartats de la pàgina web ara s'explicarà el funcionament juntament amb la vista del disseny de cada apartat. (El codi de la pàgina web, es pot trobar en annexos)

1. Home Page

Selecciona una classe

Selecciona el cicle: Infantil Primaria ESO BTX

Selecciona el curs: 1r 2n 3r 4t

Selecciona la classe: A B C

🔍 Buscar

Fig. 42: Home page de la web

En aquest apartat selecciones una classe. La web està programada per jas que els cursos i les classes s'adaptin al cicle escollit. Per exemple si selecciones btx es veuria així:

Selecciona una classe

Selecciona el cicle: Infantil Primaria ESO BTX

Selecciona el curs: 1r 2n

Selecciona la classe: A B

Buscar

Fig. 43: Adaptació dels cursos i classes al cicle

En donar-li a buscar, buscaria en la base de dades tots els nens que la classe seleccionada a la web coincideixi amb la seva classe.

2. Children List Page

← 2n BTX A

Llista d'alumnes de 2n BTX A

Número d'Alumnes totals: 5 Número d'alumnes autoritzats: 0/5 Número d'alumnes recollits: 0/5 Número d'alumnes restants: 5/5						
1		Cognoms: Armario	Nom: Joel	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú
2		Cognoms: Clemente Rodriguez	Nom: Pepito	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú
3		Cognoms: Ramal	Nom: Arnau	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú
4		Cognoms: Sevilla Rodriguez	Nom: Pau	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú
5		Cognoms: Zamora Guines	Nom: Pablo	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú

Fig. 44: Pàgina llista d'alumnes

Una vegada seleccionada la classe, apareix aquesta pàgina amb la llista de nens, aquesta pàgina en rebre un log, actualitza els tres últims camps del nen el qual un familiar seu ha sigut detectat. Quedaria d'aquesta manera:

← 2n BTX A

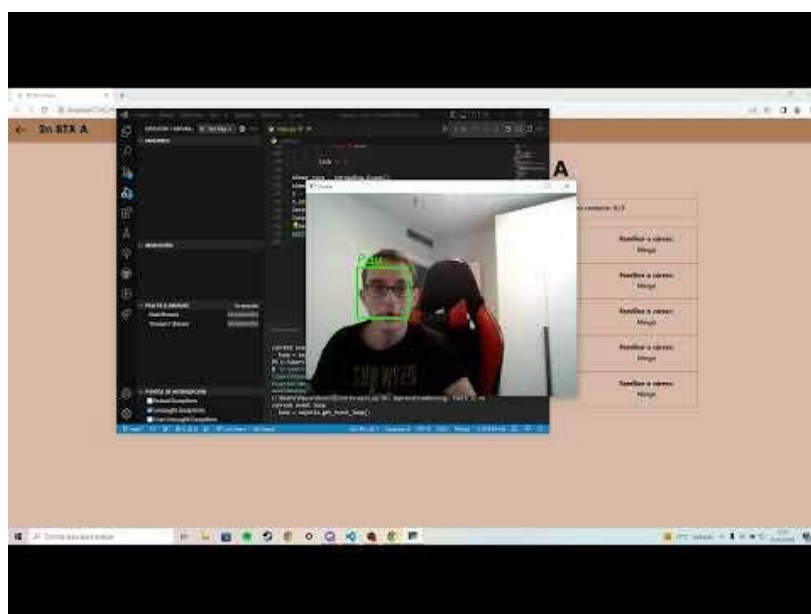
Llista d'alumnes de 2n BTX A

Número d'Alumnes totals: 5 Número d'alumnes autoritzats: 1/5 Número d'alumnes recollits: 1/5 Número d'alumnes restants: 4/5						
1		Cognoms: Armario	Nom: Joel	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú
2		Cognoms: Clemente Rodriguez	Nom: Pepito	Autoritzat a sortir ✓	Recollit? ✓	Familiar a càrrec: Pau Sevilla Rodriguez (hermano)
3		Cognoms: Ramal	Nom: Arnau	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú
4		Cognoms: Sevilla Rodriguez	Nom: Pau	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú
5		Cognoms: Zamora Oulnes	Nom: Pablo	Autoritzat a sortir ✗	Recollit? No recollit	Familiar a càrrec: Ningú

Fig. 44: Pàgina llista d'alumnes quan es detecta un familiar.

4 RESULTAT FINAL

A continuació s'adjunta un vídeo de demostració del funcionament de la web i de com detecta i indica si és conegut o desconegut un rostre:



Link: <https://youtu.be/D1QG2e-SILw>

5 POSSIBLES MILLORES

Com tot projecte, sempre hi ha aspectes que es poden millorar, aquestes són algunes de les possibles millores que s'han pensat:

1. Que hi hagi dos càmeres, llavors, una detectes els familiars que entren i l'altre els familiars amb els nens que surten. Això faria possible la detecció automàtica de si el nen ja ha estat recollit en comptes de què el professor haguí de prémer el botó quan estigui recollit.
2. Si s'hagués tingut més temps, recursos i coneixements s'hauria pogut utilitzar o fins i tot crear una IA pel reconeixement facial molt més fiable que detectes molt millors els rostres.
3. Que un familiar pugui tindre més d'un nen, es podria fer possible amb l'adaptació del codi, però caldria canviar moltes coses, si s'hagués pensat des de l'inici, no hi hauria cap inconvenient.
4. En tocar un alumne en la llista que es pugui veure més informació sobre l'alumne, les persones que el poden venir a buscar i fotografies d'aquestes.
5. Que se sincronitzés amb alèxia i que marques com a absents als nens que no hagin vingut a classe en l'última hora.
6. Ara, aquest projecte està pensat per a un sol col·legi, però mitjançant una forma de registrar-se i identificar-se es podria fer per a tots els col·legis, és una implementació factible.
7. L'algoritme detectes cares més llunyanes, però per això és necessari un ordinador més potent del que es disposava.
8. Una altra aplicació del projecte seria per sense la necessitat de gravar als nens poder controlar si algun nen entra a una zona del col·legi en la que no pot entrar.
9. Amb unes poques modificacions més també podria servir el que s'ha fet per pujar una fotografia detectar els nens que els seus pares no han acceptat els drets d'imatge i difuminar la seva cara.

CONCLUSIONS

Per acabar, una vegada fet tot el procés, tenint en compte la meua hipòtesi inicial que era: 'La intel·ligència artificial pot millorar la seguretat a l'escola'. A més a més, tenint fixant-se en les respostes que ha tingut el projecte final, es pot afirmar que la intel·ligència artificial si pot millorar la seguretat a l'escola i, d'altra banda, pot ser una eina per facilitar més la feina als professors.

S'ha pogut observar que aquest projecte ha generat alguns avantatges i també desavantatges: El primer avantatge, la més clara és la millora de la seguretat, també la facilitació pels professors, no dependre dels professors, ja que qualsevol professor amb un mòbil o tauleta pot fer aquesta tasca. Com a desavantatge és que has de tenir fotografies de les cares de tots els pares, suposa una inversió una petita inversió.

Valorant també la qualitat del projecte obtingut penso que es podria millorar, però amb coneixements que no corresponen a un nen de 2n de BTX i molt més temps, penso que el que s'ha fet en el temps que es tenia és molt satisfactori, ja que s'ha arribat a l'objectiu que es volia i inclús s'ha millorat.

Però arribar a aquesta conclusió no és una sorpresa, pel fet que s'està veient per tot arreu milers d'ampliacions on la IA resol algun problema, aquest TDR s'ha centrat en la sortida dels nens a l'escola, però pensant una mica es poden obtenir noves aplicacions on la IA ajudaria a l'escola. Com per exemple: controlar que entra o qui surt a l'escola, borra les cares dels nens que no puguin sortir a les fotografies, llegir en veu alta la pissarra pels nens que tenen dificultats de visió, i així mil idees més.

Com a valoració final del treball puc afirmar que encara que hi ha hagut diverses dificultats i que a l'inici es creia que seria molt fàcil i que en quatre setmanes ho tindria fet, la hipòtesi era totalment certa. A més a més, això ha implicat una millora personal perquè s'han après moltes coses sobre la IA i com aplicar-la en els projectes venidors.

BIBLIOGRAFIA

1. LA REALIDAD, C. [CODIFICANDOLAREALIDAD]. (2020, marzo 28). La INCREIBLE 🤖🤖 historia de la INTELIGENCIA ARTIFICIAL - Historia de la IA - Historia IA [Parte 1]. Youtube.
<https://www.youtube.com/watch?v=5eunxYo8NYk> . consulta: 15/04/2022
2. MATEU-MOLLÁ, J. (2020, febrero 24). Máquina de Turing: qué es y cómo funciona. Psicologiamente.com.
<https://psicologiamente.com/cultura/maquina-de-turing> . Consulta 15/04/2022
3. VEGA, E. (S/F). Conferencia de Dartmouth 1956. Conferencia de Dartmouth 1956. Recuperado el 5 de octubre de 2022, de
<https://dartmouthconference.wordpress.com/> . Consulta 15/04/2022.
4. ABOUT: STUDENT (COMPUTER PROGRAM). (S/F). DBpedia. Recuperado el 5 de octubre de 2022, de
[https://dbpedia.org/page/STUDENT_\(computer_program\)](https://dbpedia.org/page/STUDENT_(computer_program)) . Consulta 19/04/2022
5. FERNÁNDEZ, Y. (2017, mayo 27). Así era ELIZA, el primer bot conversacional de la historia. Xataka.com; Xataka.
<https://www.xataka.com/historia-tecnologica/asi-era-eliza-el-primer-bot-conversacional-de-la-historia> . Consulta 19/04/2022
6. INFOGRAFÍA, (ver. (2019, mayo 24). Los principales hitos en la historia de la inteligencia artificial. Gestión.
<https://gestion.pe/fotogalerias/principales-hitos-historia-inteligencia-artificial-267878-noticia/?ref=gesr> . Consulta 19/04/2022

7. GONZALEZ, A. C. L. [AprendelAconLigdiGonzalez]. (n.d.). Curso: Introducción Al álgebra lineal para machine learning. Youtube. Retrieved October 5, 2022, from <https://www.youtube.com/playlist?list=PLJjOveEiVE4DVtl4w1NWwMgPO4XszgWXa> . Consulta 15/5/2022
8. KETKAR, NIKHIL. (2017). Deep learning with python: A hands-on introduction (1st ed.). APRESS. Consula 17/6/2022
9. NO TITLE. (N.D.). Elementsofai.com. Retrieved October 5, 2022, from <https://course.elementsofai.com/es/> . Consulta 20/7/2022
10. ROVAI, M. (2018, March 12). Real-time face recognition: An end-to-end project. Towards Data Science. <https://towardsdatascience.com/real-time-face-recognition-an-end-to-end-project-b738bb0f7348> . Consulta 6/06/2022
11. Bits, C. [codificandobits]. (2020a, June 7). Reconocimiento Facial con Machine Learning. Youtube. https://www.youtube.com/watch?v=xjCrQ_XEJhl . Consulta 7/07/2022
12. NA. (2017, diciembre 12). Qué es overfitting y underfitting y cómo solucionarlo. Aprendemachinelearning.com; Aprende Machine Learning. <https://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/> . Consulta 22/04/2022
13. Bits, C. [codificandobits]. (2020b, June 14). Detección de Rostros con Machine Learning. Youtube. <https://www.youtube.com/watch?v=Y4aLYeLXAbY> . Consulta 10/07/2022
14. Bits, C. [codificandobits]. (2020c, June 21). Reconocimiento Facial con Machine Learning en PYTHON (Tutorial). Youtube. <https://www.youtube.com/watch?v=fnV6r26uBSs> . Consulta 15/07/2022

15. ROS, E. (2021, October 3). Se habla mucho de la Inteligencia Artificial... Pero, ¿en qué mejora la vida de los ciudadanos? La Vanguardia.
<https://www.lavanguardia.com/tecnologia/20211003/7760604/habla-inteligencia-artificial-que-mejora-vida-ciudadanos-brl.html> . Consulta 23/09/2022
16. IA EN SAP.PDF. (N.D.). Google Docs. Retrieved October 5, 2022, from
<https://drive.google.com/file/d/1ZlsyrHJiU0tZG3UxzbSG149HgmVm0SNt/view> . Consulta 13/08/2022
17. D11O1T, C. S. V. [DOTCSV]. (2017, November 1). ¿Qué es el Machine Learning? ¿Y Deep Learning? Un mapa conceptual | DotCSV. Youtube.
https://www.youtube.com/watch?v=KytW151dpqU&list=PL-Ogd76BhmcC_E2RjglIJZd1DQdYHcVf0 . Consulta 18/05/2022
18. COURSERA. (N.D.). Coursera. Retrieved October 5, 2022, from
<https://www.coursera.org/learn/ai-for-everyone-es/home/week/1> . Consulta 12/07/2022
19. (S/F). Cloudfront.net. Recuperado el 5 de octubre de 2022, de
https://d1wqtxts1xzle7.cloudfront.net/33824098/Research_Paper_on_Basic_of_Artificial_Neural_Network-with-cover-page-v2.pdf?Expires=1651843019&Signature=gk2piRwHG~EPFcOPy8sfOKYXQtttpMiCyeXiRNJCmAf57nw0VcJyL7pakCW1bGvNJUVHmVab18LnaB5Eymp9j9oCVsrMbXb8le-UfqOpQ5Bq9JwEIJyJhDR9NNelbtn368O4F7CTQBsVGDy5UjRAi0dhTHPsxRcCxyYbBpgQjAT0JLIHR6~wlo9V4rl2RpqtCiLM8moOS-421p7N-DlvRvq7fMSaVkp-2bfM-KXZGFZ5-ALtECoOzgTPqbn9aFM-HnMNwEixbtJwPg15tbw7hgmvhYMaa2H~RE5u2Xzbip3OVC-s~G4FPRMt47INC9kPFLReKTXVSo3Zip56jBDNnA_&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA . Consulta: 6/05/2022

20. GARCÍA, J. D. V. (2019, febrero 28). Redes neuronales desde cero (I) - Introducción. Iartificial.net. <https://www.iartificial.net/redes-neuronales-desde-cero-i-introduccion/> . Consulta: 6/05/2022
21. FUNCIÓN DE ACTIVACIÓN RELU. (2021, marzo 28). DATA SCIENCE. <https://datascience.eu/es/aprendizaje-automatico/funcion-de-activacion-relu/> . Consulta: 18/05/2022
22. ARELI, J., & BARRERA, T. (S/F). Redes Neuronales. Udg.mx. Recuperado el 5 de octubre de 2022, de http://www.cucei.udg.mx/sites/default/files/pdf/toral_barrera_jamie_areli.pdf . Consulta: 18/05/2022
23. LATINOAMÉRICA, S. (S/F). Inteligencia Artificial: ¿Qué es? Blog de Salesforce. Recuperado el 5 de octubre de 2022, de <https://www.salesforce.com/mx/blog/2017/6/Que-es-la-inteligencia-artificial.html> . consulta: 15/04/2022
24. EDITORS, C. R. (2018). Alan Turing: The life and legacy of the English computer scientist who became world war II's most famous codebreaker. Createspace Independent Publishing Platform. Consulta 15/04/2022
25. THOTH. (2016). ¿Qué es una máquina de Turing y cómo funciona? <https://formatalent.com/que-es-una-maquina-de-turing-y-como-funciona/> . Consulta 15/04/2022

ANNEX

Aquest annexos es divideixen en dos parts. La primera part programada en Python, conté tot el codi de l'algoritme de detecció de rostres, el d'augmentar les imatges i per últim els mètodes de detecció de rostres. La segona part, programada en Flutter, es a dir el disseny de la web, conté la main, la pagina inicial, la llista de nens i les rutes que segueix la web per moure's entre pàgines.

Part Python

Algoritme de detecció de rostres.

```
from dotenv import load_dotenv
load_dotenv()

import cv2
import os
import time
import asyncio
import threading
from promisio import promisify
from simple_facerec import SimpleFacerec
from supabase import create_client, Client

url: str = os.getenv("SUPABASE_URL")
key: str = os.getenv("SUPABASE_TOKEN")
supabase: Client = create_client(url, key)

updates_ids = []

@promisify
async def push_log(id_familiar, id_niño):
    ts = int(time.time() * 1000)
    supabase.table('logs').insert({'id_familiars': id_familiar,
    'id_niño': id_niño, 'time': ts}).execute()

@promisify
async def show_video(frame, z: zip):
    for face_loc, name in z:
        y1,x2,y2,x1 = face_loc[0],face_loc[1],face_loc[2],face_loc[3]
        cv2.putText(frame, name, (x1,y1-10), cv2.FONT_HERSHEY_DUPLEX, 1,
        (0,0, 200) if (name == 'desconegut') else (0,200,0),2)
        cv2.rectangle(frame, (x1,y1), (x2,y2), (0,0, 200) if (name ==
        'desconegut') else (0,200,0), 4)

    cv2.imshow('Frame', frame)

def timer(timer_runs):
```

```

while timer_runs.is_set():
    updates_ids.clear()
    time.sleep(3600)

async def start():
    #Get ids from db
    ids = dict()
    familiares = supabase.table("familiares").select('*').execute()
    for familiar in familiares.data:
        ids[familiar['nombre']] = [familiar['id'], familiar['niño']]
    #Load library and model and cam
    sfr = SimpleFacerec()
    sfr.load_encoding_images()
    cap = cv2.VideoCapture(0)

    detected_faces = {}

    tick = 0

    while True:
        ret, frame = cap.read()

        # Detect faces
        face_location, face_names = sfr.detect_known_faces(frame)

        # filter face_names to remove 'desconocido'

        video_promise = show_video(frame, zip(face_location, face_names))
        face_names = [name for name in face_names if name !=
'desconegut']
        db_promises = []

        for name in face_names:
            # check if name is a key in detected_faces
            if name not in detected_faces.copy():
                detected_faces[name] = tick

        for name, t in detected_faces.copy().items():
            if name not in face_names:
                detected_faces.pop(name)

            if tick-t == 10:
                id_familiar = ids[name][0]
                id_niño = ids[name][1]
                if id_familiar not in updates_ids:
                    db_promises.append(push_log(id_familiar, id_niño))
                    updates_ids.append(id_familiar)

        await video_promise

```

```

        key = cv2.waitKey(1)

        if key == 27:
            cap.release()
            cv2.destroyAllWindows()

            return None

        tick += 1

timer_runs = threading.Event()
timer_runs.set()
t = threading.Thread(target=timer, args=(timer_runs,))
t.start()
loop = asyncio.get_event_loop()
loop.run_until_complete(start())
timer_runs.clear()
exit(1)

```

Algoritme d'augment d'imatges

```

import cv2
import albumentations as A
import os
import face_recognition

#Define augmentor options
transform = A.augmentor = A.Compose([A.RandomCrop(width=450, height=450),
                                     A.HorizontalFlip(p=0.5),
                                     A.RandomBrightnessContrast(p=0.2),
                                     A.RandomGamma(p=0.2),
                                     A.RGBShift(p=0.2),
                                     A.VerticalFlip(p=0.5)],
                                   bbox_params=A.BboxParams(format='pascal_voc',
                                                            label_fields=[]))

print('Selecciona la carpeta:')
folder = input()

for file in os.listdir(f'./images/{folder}'):
    name, format = file.split('.')

    #Load and process image
    img = cv2.imread(f'./images/{folder}/{file}')
    img = cv2.resize(img, [450,450])
    cv2.imwrite(f'./images/{folder}/{file}', img)

    #Get face locations in order

```

```

face_loc = list(face_recognition.face_locations(img)[0])
face_loc = list(map(int, face_loc))
y1,x2,y2,x1 = face_loc[0], face_loc[1],face_loc[2],face_loc[3],
face_loc_ord = [[x1,y1,x2,y2]]

#Augmentation
i = 1
while i < 4:
    face_loc_aug = []
    list_images = [img]
    write = True

    #Get face augmentation and face location
    augmentation = transform(image=img,
bboxes=face_loc_ord)['image']
    face_loc_aug = face_recognition.face_locations(augmentation)

    #check that is not repeated
    for mat in list_images:
        if (augmentation == mat).all():
            write = False

    #Write the image
    if write == True and face_loc_aug != []:
        i += 1
        cv2.imwrite(f'./images/{folder}/{name}{i}.jpg',
augmentation)
        list_images.append(augmentation)

```

Arxiu amb els mètodes de detecció de rostres.

```

import face_recognition
import cv2
import os
import numpy as np

class SimpleFacerec:
    def __init__(self):
        self.known_face_encodings = []
        self.known_face_names = []

        # Resize frame for a faster speed
        self.frame_resizing = 0.25

    def load_encoding_images(self):

```

```

#Load images and encode with folders
for folder in os.listdir('./images/'):

    for img in os.listdir(f'./images/{folder}/'):

        imgr = cv2.imread(f'./images/{folder}/{img}')
        rgb_img = cv2.cvtColor(imgr, cv2.COLOR_BGR2RGB)

        img_encoding =
face_recognition.face_encodings(rgb_img)[0]

        self.known_face_encodings.append(img_encoding)
        self.known_face_names.append(folder)

    def detect_known_faces(self, frame):
        small_frame = cv2.resize(frame, (0, 0), fx=self.frame_resizing,
fy=self.frame_resizing)
        # Find all the faces and face encodings in the current frame of
video
        # Convert the image from BGR color (which OpenCV uses) to RGB
color (which face_recognition uses)
        rgb_small_frame = cv2.cvtColor(small_frame, cv2.COLOR_BGR2RGB)
        face_locations =
face_recognition.face_locations(rgb_small_frame, number_of_times_to_upsamp
le=3, model="cnn")
        face_encodings = face_recognition.face_encodings(rgb_small_frame,
face_locations, num_jitters=2, model="large")

        face_names = []
        for face_encoding in face_encodings:
            # See if the face is a match for the known face(s)
            matches =
face_recognition.compare_faces(self.known_face_encodings, face_encoding)
            name = "desconegut"

            # Or instead, use the known face with the smallest distance
to the new face
            face_distances =
face_recognition.face_distance(self.known_face_encodings, face_encoding)
            best_match_index = np.argmin(face_distances)
            if matches[best_match_index]:
                name = self.known_face_names[best_match_index]

            face_names.append(name)

        # Convert to numpy array to adjust coordinates with frame
resizing quickly
        face_locations = np.array(face_locations)
        face_locations = face_locations / self.frame_resizing

```

```
return face_locations.astype(int), face_names
```

Part Flutter

Arxiu Main

```
import 'package:flutter/material.dart';
import 'package:ia_web/routes.dart';
import 'package:supabase_flutter/supabase_flutter.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();

  await Supabase.initialize(
    url: 'https://xm0lryum1hfuxhfuqrty.supabase.co',
    anonKey:
      'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsI
      nJlZiI6Inhtb2xyeXVtbGhmdXhoZnVxcnR5Iiwicm9sZSI6ImFub24iLCJpYXQiOiJE2NTk4OT
      kxODQsImV4cCI6MTk3NTQ3NTE4NH0.e_f9ha8I_o3zUB09gWCsh2ab0jZzihYwKz4QT6aaLu8
      ',
    authCallbackUrlHostname: 'login-callback', // optional
    debug: true // optional
  );
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      onGenerateRoute: generateRoute,
      initialRoute: 'home',
    );
  }
}
```

Arxiu pàgina inicial

```
import 'package:flutter/material.dart';
import
'package:custom_radio_grouped_button/custom_radio_grouped_button.dart';
import 'dart:io';

class HomePage extends StatefulWidget {
  const HomePage({Key? key}) : super(key: key);

  @override
  State<HomePage> createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  bool firstTime = true;
  List<double> widths = [];
  List<String> etapa = ['Curs:', 'Cicle:', 'Classe:'];
  List<String> opciones = [
    'Selecciona el cicle:',
    'Selecciona el curs:',
    'Selecciona la classe:'
  ];
  List<List<String>> info = [
    ['Infantil', 'Primaria', 'ESO', 'BTX'],
    ['1r', '2n', '3r', '4t'],
    [
      'A',
      'B',
      'C',
    ]
  ];
  List<String> seleccion = ['Infantil', '1r', 'A'];

  @override
  Widget build(BuildContext context) {
    double width = MediaQuery.of(context).size.width;
    double height = MediaQuery.of(context).size.height;
    if (firstTime == true) {
      widths = [width * 0.15, width * 0.15, width * 0.15];
      firstTime = false;
    }
    return Scaffold(
      backgroundColor: Color(0xFFd9bba6),
      body: SingleChildScrollView(
        physics: ScrollPhysics(),
        child: Column(
          children: <Widget>[
            bigTitle(context),
```



```

    SizedBox(
      height: height * 0.1,
    ),
    //
    ListView.builder(
      physics: NeverScrollableScrollPhysics(),
      shrinkWrap: true,
      itemBuilder: (context, index) {
        return Column(
          children: [
            SizedBox(
              height: height * 0.02,
            ),
            Padding(
              padding: EdgeInsets.only(
                left: width * 0.1, right: width * 0.1),
              child: Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Expanded(
                    flex: 2,
                    child: Text(opciones[index],
                      style: TextStyle(
                        fontFamily: 'Futura',
                        fontSize: height * 0.03)),
                  ),
                  Expanded(
                    flex: 8,
                    child: CustomRadioButton(
                      defaultSelected: info[index][0],
                      width: widths[index],
                      height: height * 0.08,
                      elevation: 0,
                      absoluteZeroSpacing: false,
                      buttonLabels: info[index],
                      buttonValues: info[index],
                      buttonTextStyle: ButtonTextStyle(
                        selectedColor: Colors.white,
                        unSelectedColor: Colors.black,
                        textStyle: TextStyle(
                          fontFamily: 'Futura',
                          fontSize: height * 0.025)),
                      radioButtonValue: (String value) {
                        if (value == 'Infantil') {
                          info[1] = [
                            '1r',
                            '2n',
                            '3r',
                            '4t',

```

```

];
info[2] = [
  'A',
  'B',
  'C',
];
widths[1] = width * 0.15;
} else if (value == 'Primaria') {
  info[1] = [
    '1r',
    '2n',
    '3r',
    '4t',
    '5è',
    '6è'
  ];
  info[2] = [
    'A',
    'B',
    'C',
  ];
  widths[1] = width * 0.0855;
} else if (value == 'ESO') {
  info[1] = [
    '1r',
    '2n',
    '3r',
    '4t',
  ];
  info[2] = [
    'A',
    'B',
    'C',
  ];
  widths[1] = width * 0.15;
} else if (value == 'BTX') {
  info[1] = [
    '1r',
    '2n',
  ];
  info[2] = [
    'A',
    'B',
  ];
  widths[1] = width * 0.15;
}
setState(() {
  info;
  widths;

```

```

    });

    seleccion[index] = value;
  },
  selectedColor: Color(0xFF804000),
  unSelectedColor: Color(0xFFb17c54),
  selectedBorderColor: Colors.black,
  unSelectedBorderColor: Colors.black,
),
),
],
),
),
//
],
);
},
itemCount: 3),
SizedBox(
  height: height * 0.1,
),
searchButton(context)
],
),
),
);
}

Widget bigTitle(BuildContext context) {
  double height = MediaQuery.of(context).size.height;
  return Padding(
    padding: EdgeInsets.only(top: height * 0.1),
    child: Center(
      child: Text(
        'Selecciona una clase',
        style: TextStyle(
          fontSize: height * 0.08,
          fontFamily: 'Futura',
          fontWeight: FontWeight.bold),
      )),
  );
}

Widget searchButton(BuildContext context) {
  double height = MediaQuery.of(context).size.height;
  double width = MediaQuery.of(context).size.width;
  return SizedBox(
    width: width * 0.18,
    height: height * 0.09,

```

```

child: ElevatedButton(
  style: ElevatedButton.styleFrom(
    primary: Color(0xFFb17c54),
  ),
  onPressed: () {
    String claseArreglada =
      '${seleccion[1]} ${seleccion[0]} ${seleccion[2]}';
    Navigator.pushNamed(context, 'childrenList',
      arguments: [claseArreglada]);
  },
child: Row(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    Icon(
      Icons.search,
      size: height * 0.045,
      color: Colors.black,
    ),
    SizedBox(
      width: width * 0.01,
    ),
    Text('Buscar',
      style: TextStyle(
        fontFamily: 'Futura',
        fontSize: height * 0.045,
        color: Colors.black)),
  ],
),
),
);
}
}

```

Arxiu llista d'alumnes

```

// ignore_for_file: unused_local_variable

import 'package:flutter/material.dart';
import 'package:supabase_flutter/supabase_flutter.dart';
import 'dart:async';

final client = Supabase.instance.client;

// ignore: must_be_immutable
class ChildrenListPage extends StatefulWidget {
  ChildrenListPage({Key? key, required this.claseArreglada}) : super(key:
key);

```

```

String claseArreglada;

@override
State<ChildrenListPage> createState() =>
    // ignore: no_logic_in_create_state
    _ChildrenListPageState(claseArreglada);
}

class _ChildrenListPageState extends State<ChildrenListPage> {
    _ChildrenListPageState(this.claseArreglada);
    String claseArreglada;
    Timer? periodicTimer;
    bool timefinised = false;
    List<dynamic> logs = [];
    List<dynamic> alumnos = [];
    List<dynamic> familiares = [];
    int alumnosAuto = 0;
    int alumnosRecollits = 0;
    int alumnosRestants = 0;

    @override
    void initState() {
        super.initState();

        var time = DateTime.now().millisecondsSinceEpoch;
        periodicTimer = Timer.periodic(
            const Duration(seconds: 5),
            (timer) async {
                var futureLogs =
                    await client.from('logs').select().gt('time',
time).execute();
                logs = (futureLogs.data as List);
                for (var log in logs) {
                    final a =
                        alumnos.firstWhere((element) => element['id'] ==
log['id_niño']);
                    final f = familiares
                        .firstWhere((element) => element['id'] ==
log['id_familiares']);
                    if (alumnos[alumnos.indexOf(a)]['autorizado'] == false) {
                        alumnosAuto = alumnosAuto + 1;
                    }
                    alumnos[alumnos.indexOf(a)]['autorizado'] = true;
                    alumnos[alumnos.indexOf(a)]['familiar_autorizado'] =
                        '${f['nombre']} ${f['apellidos']} \n (${f['relacion']})';

                    // time = DateTime.now();
                }
            }
        );
    }
}

```

```

        setState(() {
          alumnos;
          timefinised = true;
        });
      },
    );
  }

  Future<List> getData() async {
    if (alumnos.isEmpty) {
      var futureAlumnos = await client
        .from('niños')
        .select()
        .eq('clase', claseArreglada)
        .execute();
      var futureFamiliares = await
client.from('familiares').select().execute();

      alumnos = (futureAlumnos.data as List);
      alumnosRestants = alumnos.length;
      familiares = (futureFamiliares.data as List);
      return futureAlumnos.data;
    } else {
      return alumnos;
    }
  }

  @override
  Widget build(BuildContext context) {
    double height = MediaQuery.of(context).size.height;
    double width = MediaQuery.of(context).size.width;
    return Scaffold(
      backgroundColor: Color(0xFFd9bba6),
      appBar: AppBar(
        leading: IconButton(
          onPressed: () => Navigator.pop(context),
          icon: Icon(Icons.arrow_back),
          color: Colors.black,
          iconSize: height * 0.035,
        ),
        backgroundColor: Color(0xFFb17c54),
        title: Text(claseArreglada,
          style: TextStyle(
            color: Colors.black,
            fontFamily: 'Futura',
            fontSize: height * 0.025)),
      ),
      body: FutureBuilder(
        future: getData(),

```

```

        builder: (context, snapshot) {
          if (alumnos.isNotEmpty) {
            alumnos.sort((a, b) =>
a['apellidos']!.compareTo(b['apellidos']!));
            return SingleChildScrollView(
              physics: ScrollPhysics(),
              child: Padding(
                padding: EdgeInsets.only(top: height * 0.04),
                child: Column(
                  children: [
                    Text(
                      "Llista d'almunes de $claseArreglada",
                      style: TextStyle(
                        fontFamily: 'Futura', fontSize: height * 0.05),
                    ),
                    SizedBox(
                      height: height * 0.04,
                    ),
                    topBar(height, width, alumnos),
                    SizedBox(
                      height: height * 0.02,
                    ),
                    ListView.builder(
                      physics: NeverScrollableScrollPhysics(),
                      shrinkWrap: true,
                      itemBuilder: (context, index) {
                        var alumno = alumnos[index];
                        return Container(
                          margin: EdgeInsets.only(
                            left: width * 0.15,
                            right: width * 0.15,
                          ),
                          padding: EdgeInsets.only(
                            top: height * 0.01,
                            bottom: height * 0.01,
                          ),
                          decoration: BoxDecoration(
                            border: Border.all(color:
Color(0xFFb17c54))),
                          child: itemAlumno(
                            alumno,
                            index,
                            height,
                            width,
                          ),
                        );
                      },
                      itemCount: alumnos.length),
                  ],

```

```

    ),
    ),
);
} else {
    return !timefinised
        ? Center(
            child: SizedBox(
                width: 200,
                height: 200,
                child: Image.asset('assets/img/lapiz.gif'))
        : Center(
            child: Text(
                "No s'ha trobat cap resultat",
                style: TextStyle(
                    fontFamily: 'Futura',
                    fontSize: height * 0.03,
                    fontWeight: FontWeight.bold,
                    color: Colors.black),
            ));
    }
},
),
);
}

Widget topBar(double height, double width, var alumnos) {
    return Container(
        height: height * 0.05,
        margin: EdgeInsets.only(
            left: width * 0.15,
            right: width * 0.15,
        ),
        decoration: BoxDecoration(border: Border.all(color:
Color(0xFFb17c54))),
        child: Row(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
                Text(
                    "Número d'Alumnes totals: ${alumnos.length}",
                    style: TextStyle(fontFamily: 'Futura'),
                ),
                SizedBox(
                    width: width * 0.01,
                ),
                Text("Número d'alumnes autoritzats:
$alumnosAuto/${alumnos.length}",
                    style: TextStyle(fontFamily: 'Futura')),
                SizedBox(
                    width: width * 0.01,

```



```

    ),
    Text(
      "Número d'alumnes recollits:
$aLumnosRecollits/${alumnos.Length}",
      style: TextStyle(fontFamily: 'Futura')),
    SizedBox(
      width: width * 0.01,
    ),
    Text("Número d'alumnes restants:
$aLumnosRestants/${alumnos.Length}",
      style: TextStyle(fontFamily: 'Futura')),
  ],
),
);
}

Widget itemAlumno(
  var alumno,
  int index,
  double height,
  double width,
) {
  return Container(
    padding: EdgeInsets.only(left: width * 0.01),
    child: Row(
      children: [
        Text((index + 1).toString(),
          style:
            TextStyle(fontFamily: 'Futura', fontWeight:
FontWeight.bold)),
        SizedBox(
          width: width * 0.02,
        ),
        CircleAvatar(
          radius: height * 0.033,
          // foregroundImage: AssetImage('img/desconocido.png'),
          backgroundImage: NetworkImage(alumno['foto'][0])),
        Expanded(
          flex: 2,
          child: Column(
            children: [
              Text('Cognoms:',
                style: TextStyle(
                  fontFamily: 'Futura', fontWeight:
FontWeight.bold)),
              SizedBox(
                height: height * 0.01,
              ),
              Text(alumno['apellidos'],

```

```

        style: TextStyle(fontFamily: 'Futura')),
      ],
    ),
  ),
  Expanded(
    flex: 2,
    child: Column(
      children: [
        Text('Nom:',
          style: TextStyle(
            fontFamily: 'Futura', fontWeight:
FontWeight.bold)),
        SizedBox(
          height: height * 0.01,
        ),
        Text(
          alumno['nombre'],
          style: TextStyle(fontFamily: 'Futura'),
        )
      ],
    ),
  ),
  Expanded(
    flex: 2,
    child: Column(
      children: [
        Text('Autoritzat a sortir',
          style: TextStyle(
            fontFamily: 'Futura', fontWeight:
FontWeight.bold)),
        SizedBox(
          height: height * 0.01,
        ),
        !alumno['autorizado']
          ? Icon(
              Icons.close,
              color: Colors.red,
            )
          : Icon(
              Icons.check,
              color: Colors.green,
            )
      ],
    ),
  ),
  Expanded(
    flex: 2,
    child: Column(
      children: [
        Text('Recollit?',

```

```

        style: TextStyle(
          fontFamily: 'Futura', fontWeight:
FontWeight.bold)),
      SizedBox(
        height: height * 0.01,
      ),
      !alumno['autorizado']
        ? Text('No recollit',
          style: TextStyle(
            fontFamily: 'Futura',
          ))
        : !alumno['recogido']
          ? ElevatedButton(
            style: ElevatedButton.styleFrom(
              primary: Color(0xFFb17c54)),
            onPressed: () {
              setState(() {
                alumnosRecollits = alumnosRecollits +
1;

                alumnosRestants = alumnosRestants - 1;
                alumno['recogido'] = true;
              });
            },
            child: Text('Recollit!'))
          : Icon(
            Icons.check,
            color: Colors.green,
          )
    ],
  )),
  Expanded(
    flex: 2,
    child: Column(
      children: [
        Text("Familiar a càrrec:",
          style: TextStyle(
            fontFamily: 'Futura', fontWeight:
FontWeight.bold)),
        SizedBox(
          height: height * 0.01,
        ),
        Text(alumno['familiar_autorizado'],
          textAlign: TextAlign.center,
          style: TextStyle(
            fontFamily: 'Futura',
          )),
      ],
    )),
  ],
),
],

```

```

    ),
  );
}

@override
void dispose() {
  super.dispose();
  periodicTimer?.cancel();
}
}

```

Arxiu de rutes

```

import 'package:flutter/material.dart';
import 'package:ia_web/pages/children_list_page.dart';
import 'package:ia_web/pages/home_page.dart';

Route<dynamic> generateRoute(settings) {
  switch (settings.name) {
    case 'home':
      return MaterialPageRoute(
        builder: (context) => HomePage(),
      );
    case 'childrenList':
      return MaterialPageRoute(builder: (_) {
        final List<dynamic> argumentos = settings.arguments;
        return ChildrenListPage(
          claseArreglada: argumentos[0],
        );
      });
    default:
      return MaterialPageRoute(builder: (context) => HomePage());
  }
}

```

AGRAÏMENTS

Per començar vull agrair aquest projecte al meu pare, ell va ser el que em va inspirar a començar a explorar el món de la programació i va despertar el meu interès per la intel·ligència artificial. També al meu tutor del TDR per guiar-me i resoldre'm els dubtes que em sorgien. A més a més, vull agrair al Pol Vallverdú per ajudar-me a resoldre alguns problemes que em van sorgir en la part de programació amb python. Per acabar, vull agrair a tota la meva família per interessar-se pel tema i donar-me ànims.