Classificació de imatges mitjançant Xarxes Neuronals Artificials i Deep Learning

Mario García Pérez

2018

Resum– Entenem com a Intel·ligència Artificial l'habilitat d'un programa o màquina d'entendre o pensar per si mateixa, utilitzant el seu entorn per aprendre i desenvolupar habilitats en base a la pràctica i experiència. Actualment, l'ús d'aquesta tecnologia, juntament amb Machine Learning i Deep Learning, ha obert un conjunt de possibilitats, inimaginables fa un temps, que poden oferir noves funcionalitats amb molt potencial. Aquest fet, impulsa a les empreses a la necessitat d'assolir els coneixements per poder millorar els seus serveis i desenvolupar nous productes que permetin aconseguir avantatge respecte als seus competidors. D'aquesta manera, una gran majoria de les grans empreses inverteixen una gran quantitat de diners en l'ús d'aquestes tecnologies i s'aprofiten d'aquest gran potencial. En aquest projecte es realitza una petita introducció al mon de les xarxes neuronals i Deep Learning realitzant un experiment de classificació de imatges on es reconeixen els dígits escrits a mà amb un molt bon percentatge d'encert. Per aconseguir-ho, s'utilitzen i es comparen diferents arquitectures de xarxa i altres propietats com funcions d'activació i d'agrupació.

Paraules clau– Xarxa, neurona, xarxa neuronal, deep learning, MNIST, Keras, TensorFlow, reconeixement de imatges, classificació de imatges, softmax, softplus, relu, softsign, dropout.

Abstract– We understand artificial intelligence as ability of a program or machine to understand or think for itself, using its environment to learn and develop skills based on practice and experience. Nowadays, the use of this technology, together with Machine Learning and Deep Learning, has opened up a set of possibilities, unimaginable some time ago, that can offer new functionalities with a very high potential. This fact drives companies to the need to get the knowledge to improve their services and develop new products that allow them to gain an advantage over their competitors. In this way, a large majority of big companies spend a big sum of money on the use of these technologies and take advantage of this great potential. In this project we make a small introduction to the world of neural networks and deep learning by doing an experiment of image classification where they recognize handwritten digits with a good percentage of success. To do this, different neural network architectures and other properties such as activation and grouping functions are used and compared.

Keywords– Network, neuron, neural network, deep learning, MNIST, Keras, TensorFlow, image recognition, image classification, softmax, softplus, relu, softsign, sigmoid, dropout

1 INTRODUCCIÓ

L'APLICACIÓ dels conceptes de xarxes neuronals en l'anàlisi d'imatges és la finalitat que presenta aquest treball. El camí a recórrer comença per provar xarxes neuronals simples amb diferents funcions d'activació per veure el comportament de cada una d'elles i la seva precisió. El següent pas consisteix en aplicar diferents tècniques d'optimització per aconseguir fer més eficients aquestes xarxes. Finalment, aplicar arquitectures "deep learning" per veure les millores que es poden arribar a assolir. Com a base del projecte, s'utilitza el conjunt de dades MNIST[1], un conjunt de dades basat en exemples de números escrits a mà molt utilitzat en casos d'iniciació a la classificació automática de imatges.

- E-mail de contacte: mario.garciap@e-campus.uab.cat
- Menció realitzada: Enginyeria de Tecnologies de la Informació
- Treball tutoritzat per: Jordi Casas Roma (Computació)
- Curs 2018/19

2 MOTIVACIÓ

Moltes vegades ens trobem amb la necessitat de digitalitzar un text escrit a mà. Fins fa ben poc, les úniques solucions a aquest problema consistien a reescriure la informació lletra a lletra o realitzar un escaneig del document. Cada tècnica té els seus punts forts i febles; copiar el text et dóna la possibilitat de reestructurar, corregir i millorar el text inicial, però per contra incrementa el temps del procés. En canvi, escanejar un document, és molt més veloç, però aquesta tècnica només permet digitalitzar el document original tal qual està escrit sense poder modificar-lo. Per aquest motiu és interessant aconseguir extreure els beneficis de cada tècnica per aplicar-les en una eina que permeti digitalitzar la informació sense gran esforç, en un temps molt reduït i tindre la possibilitat de modificar aquesta informació.

Per assolir aquest objectiu, és necessari primer introduir les tècniques d'aprenentatge automàtic per poder analitzar de manera eficaç les imatges del text escrit a mà.

3 ESTAT DE L'ART

Des de la dècada dels 50, els pioners en intel·ligència artificial ja somiaven en construir màquines complexes amb les característiques pròpies de la intel·ligència humana. Encara sembla llunyà el fet d'assolir aquest grau de IA, però realment els avenços des d'aquella època han sigut molt significatius. Actualment els termes "neural networks" o "deep learning" es troben en boca de tots els grans de la tecnologia. Empreses com Google, Apple, Amazon, són exemples clars de negocis que dediquen grans recursos en l'investigació i implementació d'aquestes noves tecnologies. Cada vegada més empreses tecnològiques s'adonen d'aquesta tendència i intenten introduir aquestes tècniques per millorar i/o competir en el seu sector.

Per exemple, Google ha creat una API [2] de visió que permet realitzar les següents tasques (entre altres) a través d'una imatge:

- Detecció d'etiquetes: Mostra etiquetes dels elements que apareixen en una imatge.
- Detecció de text: Detecta i interpreta el text que apareix en una fotografia.
- Cerca Segura: Es tracta de detectar i filtrar aquelles imatges amb contingut explícit.
- Detecció de logotips: Consisteix en la detecció dels logotips que poden aparèixer en una imatge.

Aquest software utilitza les xarxes neuronals profundes per poder extreure tota la informació valuosa de les imatges que analitza.

Com ja s'ha comentat anteriorment, en aquest projecte veurem la classificació d'imatges a partir del conjunt d'imatges MNIST. Aquest conjunt és el preferit per utilitzar com a base d'aprenentatge en el camp de les xarxes neuronals. És l'evolució/adaptació del conjunt de dades NIST. MNIST consta d'un conjunt de 70.000 exemples de dígits escrits a mà, per aproximadament 500 persones diferents, dividits en 60.000 per entrenar la xarxa neuronal i 10.000 reservats com a conjunt de proves.

En les últimes dues dècades, s'han realitzat diferents experiments amb diferents tècniques i mètodes per aconseguir una taxa d'error mínim sobre aquest conjunt de dades. Per exemple el 1998 s'aconseguia mitjançant classificadors lineals amb una sola capa, una taxa d'error del 12%, la qual podia ser reduïda al 8.4% utilitzant un procés anomenat *"deskewing"*[3], que permetia modelar la imatge perquè la computadora aconseguís una millor interpretació.

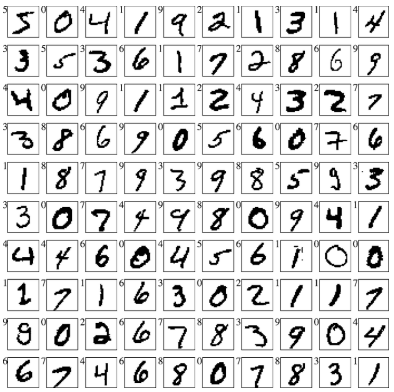

Fig. 1: Digits MNIST. (Font: neuralnetworksanddeeplearning.com)

A causa d'aplicar diferents tècniques de tractament d'imatge juntament amb estructures de xarxes més complexes; com les xarxes convolucionals, actualment, s'aconsegueix una taxa d'error de 0.23% [4] utilitzant distorsions elàstiques i amb processos de normalització. Però fins i tot s'han registrat resultats de 0.21%, molt superior a les taxes inicials i als resultats que ofereim els mateixos humans.

4 OBJECTIUS

Els objectius principals que es pretenen acoseguir en aquest projecte son els seguents:

- Comprendre els conceptes, el funcionament i l'abast de les xarxes neuronals partint de models més bàsics fins a xarxes neuronals convolucionls, que representen l'estat de l'art en l'actualitat.
- Analitzar el funcionament i el rendiment de diferents arquitectures de xarxes neuronals, incloent xarxes monocapa, multicapa i convolucionals.
- Comprovar l'evolució del grau d'eficiencia que ens ofereix una xarxa neuronal modificant técniques i estructures.
- Construir un model predictiu eficaç en la classificació de digits escrits a mà, obtenint valors de precisió propers als actuals nivells en l'estat de l'art.

A la figura 2 es mostra un esquema del model a assolir.

5 FASES DEL PROJECTE

En l'apèndix d'aquest document es pot trobar un diagrama de Gannt amb un resum dels processos a realitzar durant tot el projecte i una taula amb la planificació més detallada d'aquestes tasques.

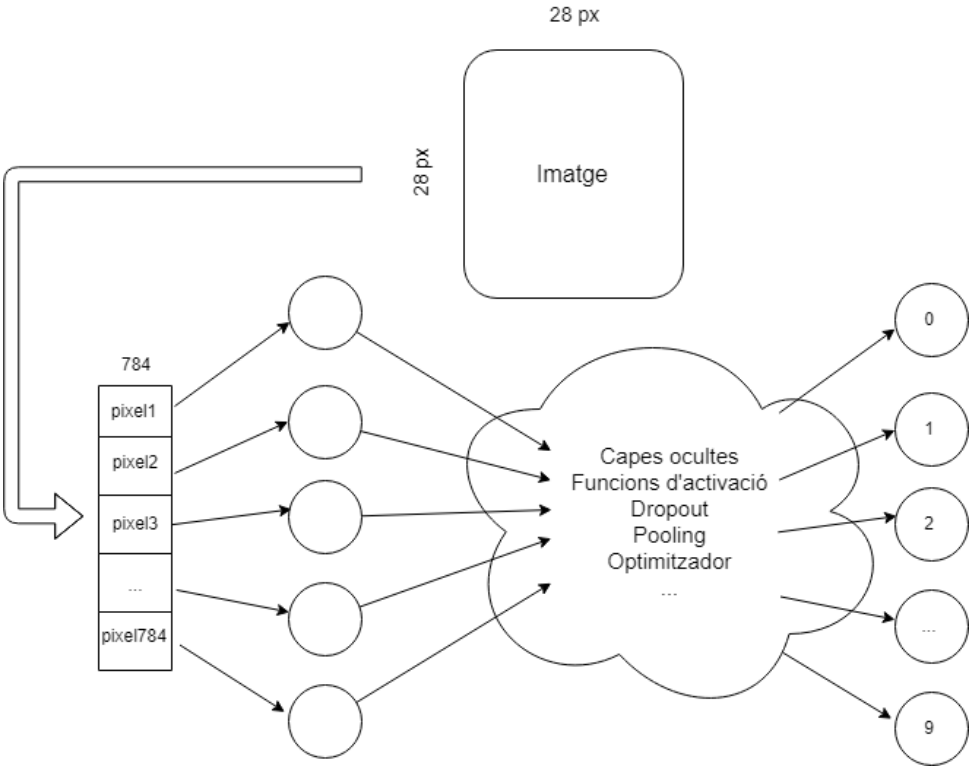

Fig. 2: Esquema conceptual inicial.

6 CONCEPTES BÀSICS

Mitjançant la realització del present projecte, es pretén recórrer el camí bàsic que composen les xarxes neuronals. L'inici ve marcat pel concepte de **neurona**. En aquest context, una neurona és una unitat de càlcul que intenta modelar el comportament d'una neurona cerebral. Les xarxes neuronals focalitzen la seva essència en elles. A aquesta neurona se l'associa una entrada i una sortida. Tal i com es pot veure en la figura 3, el resultat del càlcul en una neurona consisteix en realitzar una suma ponderada de les entrades assignades, tot seguit de l'aplicació d'una funció no lineal.

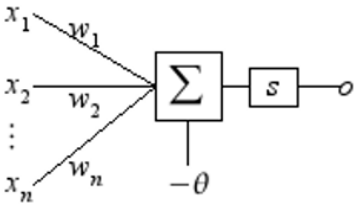

Fig. 3: Estructura d'una neurona

En aquest cas:

- S correspon a la funció d'activació
- *x* correspon al valor de l'entrada
- *w* correspon al pes assignat a la connexió
- θ correspon al valor Umbral de la neurona
- *o* correspon a la sortida de la neurona

D'aquesta manera podem dir que:

$$o = S(\omega_1 x_1 + \omega_2 x_2 + \omega_3 x_3) - \theta$$

Al connectar la sortida d'una neurona amb l'entrada d'una altra, aconseguim modificar la seva estructura i crear així més capes per formar una xarxa. Aquest concepte es representa tal i com es pot observar en la figura 4:

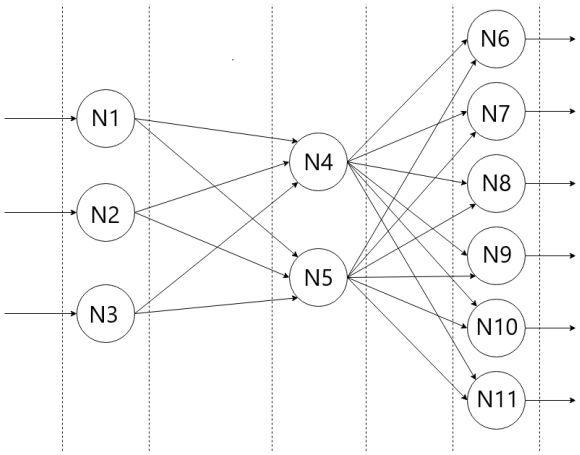

Fig. 4: Estructura d'una xarxa

Com ja s'ha indicat anteriorment, un dels objectius del present projecte és aconseguir un valor de precisió el més proper al cent per cent possible. Això ve donat per la configuració ideal de les variables acabades de presentar.

L'activitat consisteix en generar una xarxa neuronal en la que la sortida generada sigui el valor del número que la xarxa ha obtingut com a entrada però en format d'imatge. Es a dir, la xarxa ha d'endevinar quin és el número escrit a

mà. Quants més encerti, obtindrà un valor de precisió més elevat.

Per aconseguir aquests valors, es diferencien dues fases de modelització:

- Fase d'entrenament: en aquesta fase s'utilitza una part del conjunt de dades (MNIST en aquest cas) per determinar els pesos que defineixen el model de la xarxa neuronal. Aquests pesos es calculen de manera iterativa, amb l'objectiu de minimitzar l'error obtingut entre la sortida resultant de la xarxa i el valor de sortida esperat o desitjat. Els *epochs* representen el nombre d'iteracions en que l'entrenament recorre la xarxa.

  Un altre concepte que s'utilitza en l'entrenament consisteix en la retropropagació. Backpropagation [5] és un mètode de càlcul de pesos que s'utilitza en algoritmes d'aprenentatge i que ofereix molt bons resultats gracies a re-ajustar els pesos dels nodes en direcció contraria després d'haver recorregut la xarxa de principi a final.

  Durant aquesta fase, el model pot patir un sobreentrenament, de manera que s'ajusta massa a les particularitats del patró d'entrenament. En aquest cas es perdria l'habilitat de generar el seu aprenentatge en nous casos amb altres dades.

- Fase de prova: per evitar aquest cas, s'utilitza un segon grup de dades diferents als d'entrenament per permetre controlar aquest procés d'aprenentatge.

  Aquest grup de dades s'utilitza per veure com reacciona el model davant dades que no ha analitzat anteriorment i es pot saber quin és el seu grau de precisió i el seu grau d'error.

Per realitzar aquest experiment, s'utilitza la llibreria de Python Keras.[6] Keras proporciona de manera neta i simple la creació de models de xarxes neuronals, funcionant com a nivell superior d'altres llibreries com Theano, CNTK o TensorFlow[7]. En aquest cas, la capa baixa utilitzada és aquesta última, TensorFlow.

El primer pas en el desenvolupament tècnic del projecte consisteix en crear una xarxa neuronal bàsica d'una sola capa. Aquesta vindria representada per 784 unitats d'entrada, que corresponen al nombre de píxels que conté la imatge del model de dades, i 10 sortides, que representen els deu possibles números [0 - 9] al que pot representar la imatge d'entrada.

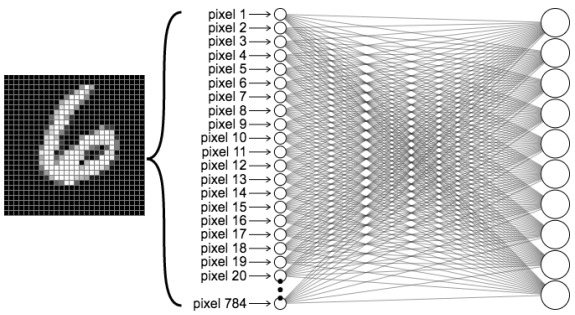

Fig. 5: Estructura d'una xarxa bàsica per MNIST. (Font: ml4a.github.io)

A partir d'aquí, modificant diferents paràmetres, s'obtenen valors de sortida diferents.

El primer concepte que tindrem en compte és la funció d'activació. En aquest cas, Keras ens ofereix de manera senzilla, simplement indicant el nom de la funció, l'aplicació dels diferents processos que realitza cada funció.

A continuació s'expliquen algunes d'aquestes **funcions d'activació** i els resultats obtinguts partint del mateix model d'entrenament inicial.

- Softmax: Aquesta funció representa una generalització de la funció logística de tal manera que comprimeix un vector de valors reals arbitraris, convertint-lo en un vector en rang [0, 1]. La funció Softmax s'utilitza sovint com a capa final en els classificadors basats en xarxes neuronals.

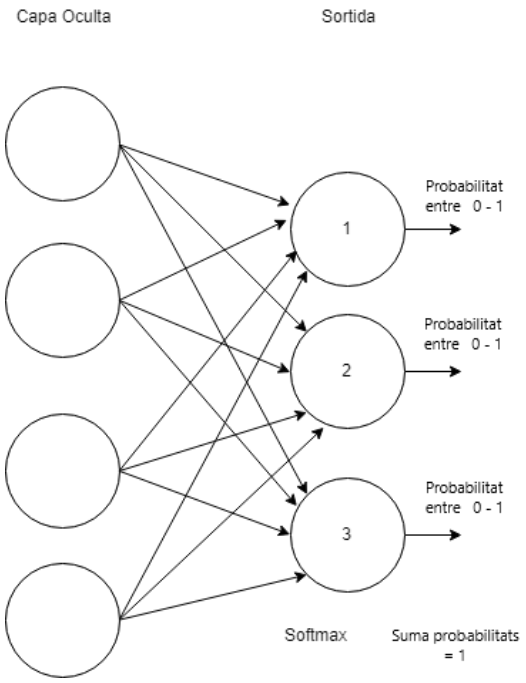

Fig. 6: Estructura d'una xarxa bàsica amb funció d'activació Softmaxt

- ReLU: L'Uniat Lineal Rectificadora, és una funció d'activació coneguda com a funció rampa definida per la formula:

$$f(x) = max(0, x)$$

  Les seves aplicacions més comuns son en visió per computadors y reconeixement de veu utilitzant xarxes neuronals profundes.

- SoftPlus: és una funció d'activació que s'aproxima a la funció ReLU, però té un comportament més suau. La seva formula ve donada per:

$$f(x) = ln(1 + e^x)$$

  A continuació, en la figura 7, es mostra les diferencies entre aquestes dos funcions d'activació:

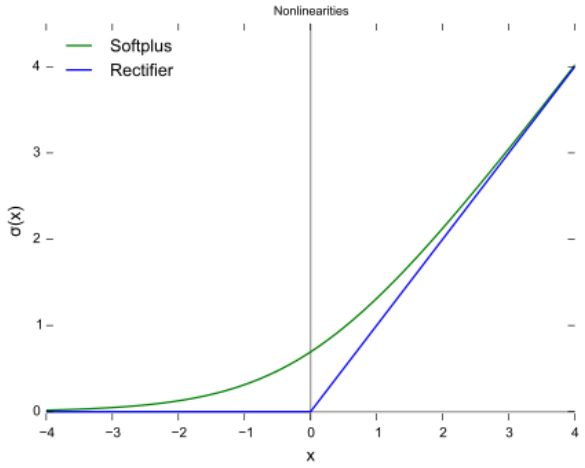

Fig. 7: Gràfic comparatiu entre funcions. (Font: wikipedia.org/wiki/Rectifier_(neural_networks))

- Sigmoid: La formula que representa aquesta funció és la següent:

$$f(x) = \frac{1}{1 + e^{-x}}$$

  Aquesta funció es representa gràficament amb forma de S, la cual és una funció real de variable real diferenciable. A més a més, igual que la funció Softmax, converteix la sortida en un vector de rang [0, 1].

- SoftSign: Aquesta funció d'activació converteix els valors en un vector de rang [-1, 1]. La seva formula ve donada per:

$$f(x) = \frac{x}{1 + |x|}$$

A continuació es mostra una taula comparativa amb els valors obtinguts durant l'execució d'aquestes tècniques utilitzant el mateix model d'entrenament:

	Test loss	**Test accuracy**
SOFTMAX	0.260964198371768	0.9278
SIGMOID	0.260946302357316	0.9277
SOFTSIGNS	7.786724453735352	0.1118
SOFTPLUS	0.262396398997306	0.9273

La taula mostra valors semblants en les funcions Softmax, Sigmoid i Softplus, molt superiors a les altres funcions, on la taxa d'error és molt elevada. Però això no implica que els altres models s'hagin de descartar, només que no son eficaços en aquest model concret.

7 Desenvolupament Tècnic

7.1 Escenari

Un cop explicat aquests termes i exposats els resultats que proporcionen, el present document continua amb l'exposició del models creats per realitzar els experiments per demostrar les millors tècniques per obtenir resultats òptims.

Primer de tot, cal destacar l'entorn dels experiments:

- CPU: I7-6700K
- GPU: GTX 1050 ti
- RAM: 8 GB DDR4
- Ubuntu 18.01.1
- Python: Tal i com ja se sap dins del món de d'intel·ligència artificial, Python és un dels llenguatges de programació més utilitzats. És un llenguatge multiplataforma, orientat a objectes i fàcil d'interpretar per gràcies a la seva sintaxis. Ha estat escollit gràcies a la seva quantitat de llibreries, tipus de dades y funcions incorporades en el propi llenguatge, que ajuden a realitzar moltes tasques sense necessitat de haver-les de programar de zero. D'aquesta manera, s'ha aprofitat de TensorFlow per poder realitzar tota la gestió de les xarxes.
- TensorFlow: Es tracta d'una biblioteca de codi obert que es basa en un sistema de xarxes neuronals. És l'encarregada de gestionar, des de la capa més baixa, les xarxes creades. Proporciona la construcció, entrenament, i avaluació d'aquestes.

  La biblioteca TensorFlow ha estat escollida en aquest projecte a causa de les seves capacitats actuals, el creixement que està patint últimament, tant en desenvolupament com en usuaris, i finalment pel potencial que té gràcies a que es desenvolupada per la multinacional Google.
- Keras: Keras es considerada una capa superior de TensorFlow, Theano o CNTK. Tècnicament, es una llibreria de Python que proporciona, d'una manera simple i eficient, la creació de models de Xarxes Neuronals.

  Keras ha estat escollida per les seves característiques; la facilitat d'us per sobre de TensorFlow.
- Spyder3: (Scientific Python Development Environment) es un potent entorn de desenvolupament interactiu pel llenguatge Python. Proporciona funcions avançades de edició, test i depuració que afavoreixen i faciliten la part pràctica d'aquest projecte, sent el pont entre el codi i l'usuari.

7.2 Xarxa tradicional bàsica

Un cop vist l'escenari utilitzat durant el projecte, es procedeix a documentar els experiments realitzats, els problemes detectats i els resultats extrets d'aquests experiments.

El primer pas realitzat consisteix en carregar el model de dades MNIST explicat anteriorment per a poder tractar-lo. Keras facilita aquesta part gràcies a que ja incorpora el conjunt de dades MNIST i només cal importar-lo i guardar-lo en les variables de d'entrenament i test. Un cop fet, només cal formatar correctament les dades carregades d'aquest conjunt.

En aquest punt, tenim 60.000 mostres de dígits del 0 al 9 escrits a mà, introduïts en una variable llesta per ser tractada per la xarxa que es crea a continuació.

El següent pas consisteix en la creació del model de la xarxa. Primerament, s'ha creat una xarxa amb estructura

bàsica on les entrades connecten directament amb la sortida de 10 neurones (una per valor del 0 al 9).

A continuació es mostra l'exemple indicat:

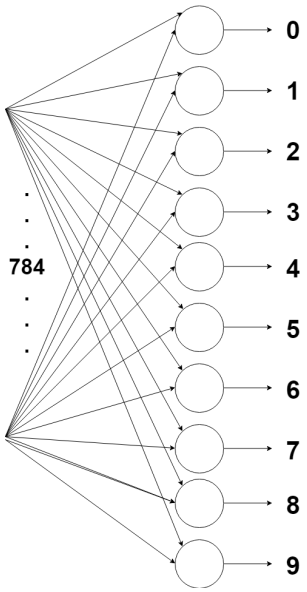

Fig. 8: Estructura d'una xarxa bàsica

Aquesta estructura es creada de la següent manera:

```
model = Sequential()
model.add(Dense(10,
          activation='softmax',
          input_shape=(784,)))
```

El model creat, representa un model seqüencial amb 784 entrades per una capa de 10 sortides i una funció d'activació Softmax presentada anteriorment.

Fins aquest punt, podríem dir que s'ha aconseguit construir una petita xarxa neuronal capaç d'identificar nombres del 0 al 9 amb un percentatge d'encert prou convincent, però simplement serveix com a punt de partida.

La figura 9 mostra els resultats obtinguts mitjançant aquesta xarxa. Primerament es poden observar dos gràfics que representen els valors *accuracy* i *loss* respecte els *epochs*, que es resumeix en el percentatge d'encert respecte cada pasada per tot el model de dades i el valor d'error respecte cada iteració del model.

Finalment s'indiquen els millors valors assolits durant l'execució.

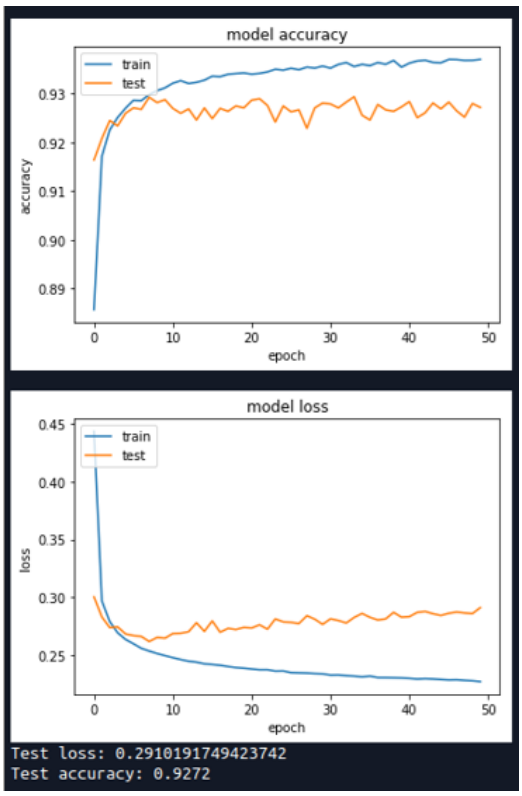

Fig. 9: Resultats d'execució d'una xarxa bàsica.

Aquests resultats ens ensenyen que per aquest model, no és eficaç un alt nombre d'*epoch*, ja que els valors òptims es troben entre les 10 primeres repeticions.

D'aquí en endavant, es pretén cercar, entendre i utilitzar tècniques que proporcionin més eficiència al model.

Per assolir aquest objectiu, es pretén modificar tant l'estructura de la xarxa com les seves propietats.

El primer escenari modificat ha constat en augmentar el nombre de capes de la xarxa, introduint capes intermitges. En aquest nou model que es presenta a continuació, es prenen 784 valors d'entrada connectats a 128 nodes amb una d'activació Softplus. Aquesta capa, torna a connectar amb un altre de 128 nodes amb un altre funció d'activació Softplus que finalment, connecta amb una capa de 10 sortides mitjançant un altre funció d'activació Softplus.

```
model = Sequential()

model.add(Dense(128,
          activation='softplus',
          input_shape=(784,)))

model.add(Dense(128,
          activation='softplus'))

model.add(Dense(10,
          activation='softplus'))
```

El resultat d'aquesta modificació i per tant, d'aquest nou model, és el següent:

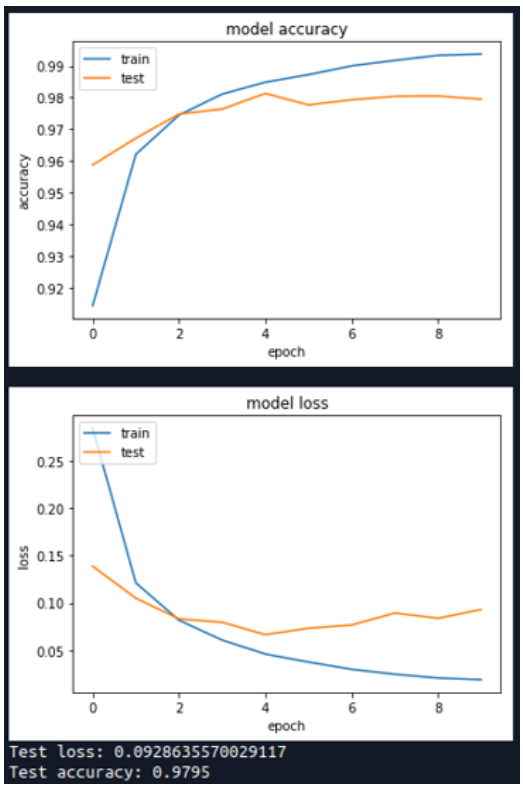

Fig. 10: Resultats d'execució de la primera xarxa amb capes ocultes.

Com es pot observar, s'aconsegueix una millora important respecte el model anterior, factor indicatiu de que el model és més eficient aconseguint un valor de 97,95 % en la precissió d'aquest.

Un cop arribat a aquest punt, el plantejament ha consistit en utilitzar un mètode que permeti fer probes repetidament del model amb diferents paràmetres per tal d'obtenir la millor combinació possible. Això ha estat gracies a GridSearch.[8]

GridSearch és una llibreria que incorpora Keras capaç d'oferir i simplificar combinacions "d'hiperparametres", com els *epochs*, el *batch size*, optimitzadors o les funcions d'activació, repetint l'entrenament i test del model i emmagatzemant els resultats obtinguts, de tal manera que al finalitzar l'execució, indica quin ha sigut el millor resultat (best_score_) i amb quina combinació (best_params_).

7.3 Xarxa tradicional amb millor resultats

Després de molts entrenaments, tests i modificacions, hem aconseguit un model que obté uns resultats molt positius. Aquest model consta d'una estructura similar a les explicades prèviament però afegint-hi un nou concepte, el Dropout.

El Dropout [**?**] és una tècnica que consisteix en establir aleatòriament alguns valors de l'entrada de la xarxa a 0. Aquest mètode és utilitzat amb la finalitat de evitar que les xarxes neuronals pateixin "sobre-entrenament". El "sobre-entrenament", és un possible efecte d'una xarxa en que l'algoritme d'aprenentatge s'acostuma a les dades d'entrada i coneix el resultat desitjat. La xarxa neuronal ha de ser capaç d'assolir un estat on sigui capaç de predir el resultat en altres casos a partir de lo après durant l'entrenament. Però malauradament, quan un sistema s'entrena massa, l'algoritme d'aprenentatge pot quedar ajustat a unes certes característiques específiques de les dades d'entrenament, allunant-se de l'objectiu final d'aquest.

Aquest cas es pot detectar ja que mentre els valors d'eficiència dels entrenaments va augmentant, els de test van disminuint.

Les figures 10 i 11, mostren l'exemple de models creat on els valors d'entrenament milloren mentre que els de test decauen a mida que augmenten els *epochs*.

Tornant al camí principal, després d'ajustar tots els valors necessaris, el model resultant ha sigut el següent:

```
model = Sequential()

model.add(Dense(512,
          activation='relu',
          input_shape=(784,)))

model.add(Dropout(0.2))

model.add(Dense(512,
          activation='relu'))

model.add(Dropout(0.2))

model.add(Dense(10,
          activation='softmax'))
```

Aquest model, utilitza 784 valors d'entrada que corresponen als píxels de cada imatge del conjunt de dades MNIST, aquests valors son multiplicats per un pes concret abans de ser sumats entre si. Un cop sumats, s'aplica la funció d'activació Relu multiplicant-lo pel seu valor.

Tot seguit, la funció Dropout anul·la un percentatge de les sortides reinicialitzant el valor a 0 i evitant així el "sobre-entrenament". La següent capa és similar, ja que conserva la mateixa funció d'activació però utilitza els 512 valors de sortida de la capa anterior i torna a extreure uns altres 512 valors, dels quals un percentatge tornen a ser inicialitzats a 0 a causa de la funció Dropout. Finalment, l'ultima capa obté les 512 sortides anteriors com a entrada i, mitjançant la seva funció d'activació Softmax, extreu 10 sortides on els valors son diferents.

Finalment, s'avalua quina de les sortides té un pes més significatiu i es prediu que la sortida amb aquest valor és el dígit que s'ha utilitzat com a entrada.

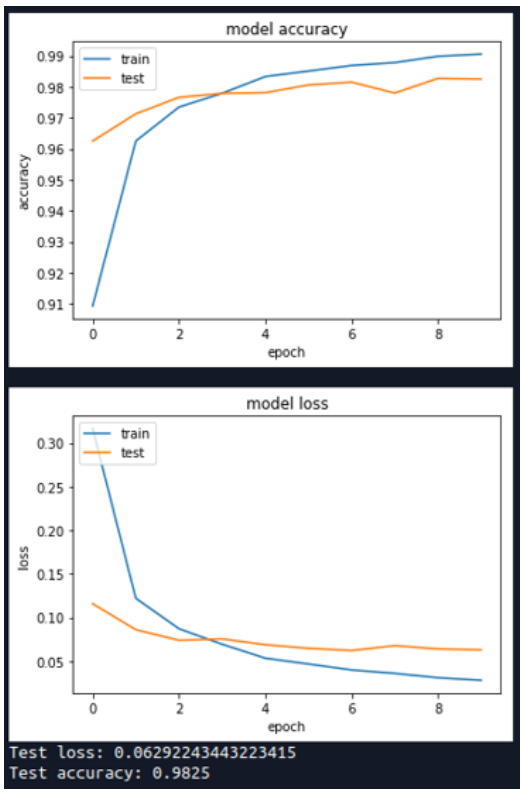

Fig. 11: Resultats d'execució de la xarxa amb dropout.

7.4 Introducció a les xarxes convolucionals

Deixant de banda les xarxes tradicionals o *"fully Connected"*, passem a les xarxes neuronals convolucionals. Aquestes son unes xarxes amb una arquitectura especial adaptada específicament per la classificació d'imatges, de tal manera que guanyen pes en aquest projecte.

Gràcies a les seves propietats, aquestes son ràpides d'entrenar en el context del projecte i facilita els seus bons resutats.

A continuació es detallen les tres grans diferencies entre les xarxes utilitzades fins el moment i les xarxes convolucionals recentment explicades.

- *Shared Weights*: Aquest concepte indica que les xarxes convolucionals no utilitzen valors de pesos diferents en les capes ocultes com han fet fins ara les xarxes tradicionals que s'han descrit durant el projecte, sinó que aquestes utilitzen uns valors de pesos idèntics durant cada una de les capes ocultes, de tal manera que totes les neurones de cada capa detecta la mateixa característica en la imatge, però cada una en una posició diferent.

- *Pooling Layers*: La funció d'aquesta característica es pot definir com una agrupació en la sortida amb la finalitat de simplificar aquesta informació. Algunes de les millores que s'obtenen mitjançant aquesta tècnica son les següents:

  – Preveu el sobre-entrenament, ja que al haver-hi menys informació espacial es simpli-fiquen els paràmetres.

  – Al reduir el tamany d'informació, augmenta el rendiment de càlcul.

  Cal destacar també, algunes de les funcions d'agrupació més utilitzades:

  – *Max-pooling*: Aquesta funció simplement selecciona el valor màxim del conjunt d'entrada. L'aplicació d'aquesta funció en una capa d'agrupació, permet que la xarxa conegui si una característica concreta es troba en algun lloc de la imatge.

  – *Average-pooling*: Aquesta funció, en comptes de seleccionar el valor màxim del conjunt, calcula una mitja entre tots els seus valors. Aquest mètode extreu altres característiques però normalment no detecta característiques extremes tal i com fa max-pooling. A continuació, es mostra la figura 12 que indica les diferencies d'aquests dos mètodes:

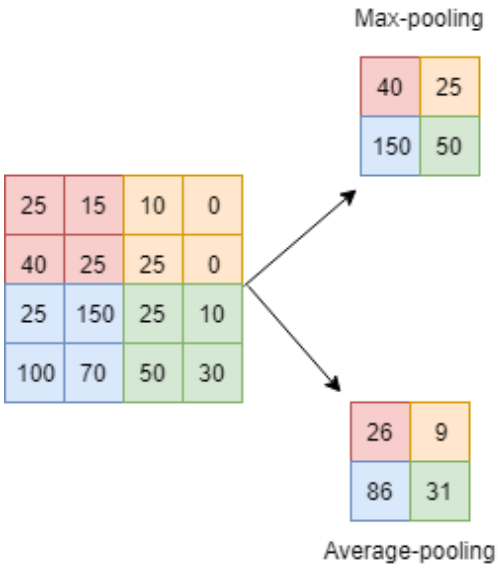

Fig. 12: Comparació de les funcions *max-pooling* i *average-pooling*.

  – *L2-pooling*: Aquesta funció calcula l'arrel cuadrada de la suma dels cuadrats dels valors d'activació. La finalitat es molt semblant a les anteriors, agrupar i reduir la informació de la capa convolucional.

- *Local Receptive Fields*: Fins ara hem vist com tots els models creats, utilitzaven com a entrada cada un dels píxels de la imatge inicial (784). Les xarxes convolucionals connecten els píxels en funció de la seva posició en la imatge, aprofitant la relació entre ell i els píxels veïns ja que son els que tenen més valor pel primer. D'aquesta manera, s'agrupen els píxels propers creant un únic resultat que representa a quest conjunt.

  A continuació es mostra un exemple on un grup de les neurones de la primera capa connecten amb una única neurona de la següent, representativa del primer conjunt.

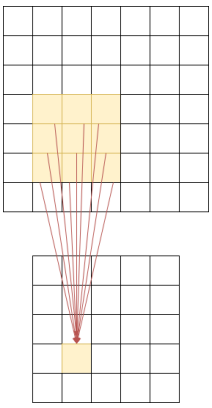

Fig. 13: Exemple teòric Local receptive fields.

7.5 Arquitectura de la xarxa convolucional

Finalment, després d'entrenaments, tests i modificacions, s'ha aconseguit generar un model de xarxa capaç de proporcionar un resultat d'encert molt pròxim al cent per cent.

Aquest model utilitza funcions convolucionals combinades amb tècniques com max-pooling, Dropout amb diferents percentatges d'eliminació, diferents funcions d'activació i un optimitzador "Adelta".

A continuació es mostra el model utilitzat per aquesta xarxa:

```
model = Sequential()
model.add(Conv2D(32,
          kernel_size=(3, 3),
      activation='relu',
      input_shape=input_shape))

model.add(Conv2D(64, (3, 3),
          activation='relu'))
model.add(MaxPooling2D
          (pool_size=(2, 2)))

model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128,
          activation='relu'))

model.add(Dropout(0.5))
model.add(Dense(num_classes,
          activation='softmax'))

model.compile(loss
          =keras.losses.
          categorical_crossentropy,
          optimizer
          =keras.optimizers.Adadelta(),
          metrics=['accuracy'])
```

7.6 Resultats de la xarxa convolucional

Com ja s'ha indicat, el model anterior presenta un gran valor d'encert, el qual ha sigut el seguent:

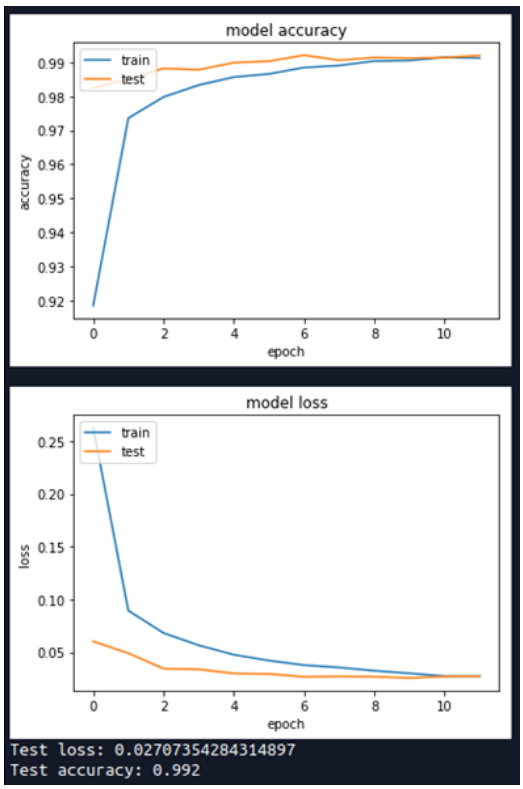

Fig. 14: Resultats d'execució de la xarxa neuronal convolucional.

Un 99,2 % és un percentatge d'encert molt alt, el qual, encara que no és un valor molt allunyat de l'obtingut mitjançant les xarxes tradicionals, significativament és un valor molt llunyà, ja que supera els valors màxims teòrics obtinguts per experts mitjançant aquestes xarxes tradicionals.

A mode comparatiu, a continuació es mostra una taula on s'indiquen alguns dels resultats obtinguts mitjançant diferents models de xarxes.

Xarxa	Test loss	Test accuracy
bàsica (Fig. 9)	0.29101	0.9272
amb capes ocultes (Fig. 11)	0.09286	0.9795
amb molts epochs	0.10129	0.9824
amb combinacions de funcions d'activació	0.09003	0.9835
amb combinacions aleatories de funcions d'activació	7.7867	0.1118
amb dropout i diferents valors d'entrada	0.06403	0.981
amb dropout (Fig. 12)	0.06292	0.9825
Convolucional (Fig. 13)	**0.02707**	**0.9920**

8 Conclusions

Com hem pogut observar, les xarxes neuronals proporcionen un gran potencial davant la resolució de problemes de predicció i classificació. En aquest cas, s'ha construït un model capaç de resoldre un problema relativament senzill per un ésser humà, però assolint amb un percentatge d'encert molt elevat. Donat aquest resultat tant elevat, seria molt

interessant comparar-lo amb el percentatge d'encert de diferents éssers humans sobre un model nou.

Tenint en compte que el problema resolt es considera senzill e introductori, la seva resolució il·lumina la idea sobre la complexitat dels problemes que es poden arribar a solucionar mitjançant l'aplicació de les xarxes neuronals en qualsevol context.

Tot i així, les aplicacions que es poden extrapolar del model generat podrien arribar a ser molt útils i amb una bona acceptació comercial.

En quant als objectius fixats al començament del projecte, s'assoleix completament el propòsit principal de generar un model basat en xarxes neuronals capaç d'interpretar dígits escrits a mà amb una eficiència molt elevada.

REFERÈNCIES

[1] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010.

[2] Google, "Google cloud vision api," 2015.

[3] D. Ghosh and A. Wan, "Deskewing," 2013.

[4] U. M. Ciresan, D. and J. Schmidhuber., "Multi-column deep neural networks for image classification," 2012.

[5] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*, pp. 65–93, Elsevier, 1992.

[6] F. Chollet *et al.*, "Keras," 2015.

[7] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[8] Y. B. James Bergstra, "Random search for hyper-parameter optimization," 2012.

APÈNDIX

A continuació es mostren les figures que fan referencia a la planificació del projecte.

TAULA 1: PLANIFICACIÓ DE TASQUES

Setmana 1	Reunió amb el tutor per realitzar l'anàlisi dels requeriments del treball. Preparació de l'entorn de treball.
Setmana 2, 3	Estudi sobre el tema del projecte. Anàlisi de sistemes existents. Planificació de les tasques per dur a terme el projecte. Construcció d'una primera xarxa neuronal bàsica.
Setmana 4	Estudi sobre el tema del projecte. Entrega del 1er informe.
Setmana 5	Estudi sobre el tema del projecte. Aplicació de diferents funcions d'activació i organització dels resultats obtinguts.
Setmana 6, 7	Estudi sobre el tema del projecte. Aplicació de diferents estructures de xarxes i tècniques d'optimització.
Setmana 8	Organització i valoració dels resultats. Finalització de xarxes tradicionals o "fully Connected".
Setmana 9	Finalitzar documentació i entrega del 2n informe.
Setmana10, 11	Estudi sobre el tema del projecte. Primeres proves amb xarxes convolucionals.
Setmana 12	Estudi sobre el tema del projecte. Implementació de xarxes convolucionals.
Setmana 13	Anàlisi dels resultats obtinguts.
Setmana 14	Comparació de resultats. Preparació documentació del 3er informe.
Setmana 15	Entrega del 3er informe.
Setmana 16, 17, 18	Redactar memòria final.
Setmana 19	Redactar memòria final. Preparar presentació.
Setmana 20	Entrega del projecte.

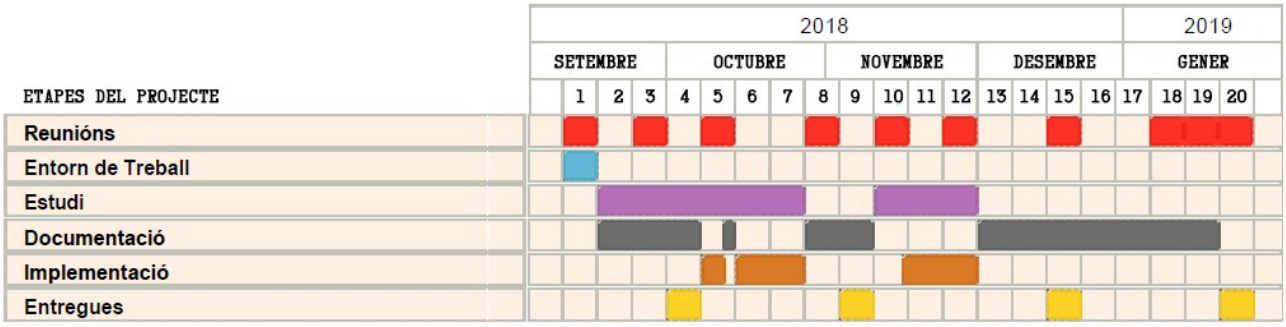

Fig. 15: Diagrama de Gannt