UNIVERSITAT DE BARCELONA
Facultat de Matemàtiques i Informàtica

GRAU DE MATEMÀTIQUES

Treball final de grau

LES XARXES NEURONALS DE PROPAGACIÓ CAP ENDAVANT. UNA APROXIMACIÓ MATEMÀTICA

Autor: Raül Sánchez Albaladejo

Director: Eloi Sans Gispert
Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, 18 de juny de 2021

Abstract

In this work we describe what feedforward neural networks are and how they are used. We explain the elements that make them up: layers, depth, weights, biases, learning rate and activation function.

Then we see that feedforward neural networks are universal approximators of functions under certain conditions. We study two different ways to prove this. On the one hand, the Kolmogorov-Sprecher pathway tells us that feedforward neural networks with three layers, n components in the first layer, $2n + 1$ nodes in the second layer, and m nodes in the last layer are universal approximators of continuous functions from $\mathbb{R}^n$ a $\mathbb{R}^m$ as long as the activation function is monotonically increasing and class $Lip[\frac{ln2}{ln(2N+2}]$. On the other hand, in the second pathway we see that feedforward neural networks are universal approximations of any measurable function as long as the activation function of the neural network is a squashing function.

Finally we explain how to determine the different elements that configure the feedforward neural networks. We define the cost function. We explain that by minimizing the cost function by the stochastic gradient descent method and the learning rate we can calculate the weights and biases. At the end we study different activation functions and see how they affect neural networks also explaining the vanishing gradient.

Resum

En aquest treball describim qué són les xarxes neuronals de propagació cap endavant i com es fan servir. Expliquem quins són els elements que les configuren: capes, profunditat, pesos, biaixos, taxa d'aprenentatge i funció d'activació.

Després veiem que les xarxes neuronals de propagació cap endavant són aproximadors universals de funcions sota certes condicions. Estudiem dues formes diferents de demostrar-ho. Per una banda la via Kolmogorov-Sprecher ens diu que les xarxes neuronals de propagació cap endavant amb tres capes, n components a la primera capa, $2n + 1$ nodes a la segona capa i m nodes a l'última capa són aproximadors universals de funcions contínues de $\mathbb{R}^n$ a $\mathbb{R}^m$ sempre que la funció d'activació sigui monòtona creixent i de classe $Lip[\frac{ln2}{ln(2N+2)}]$. Per altra banda a la segona via veiem que les xarxes neuronals de propagació cap endavant són aproximadors universals de qualsevol funció mesurable sempre que la funció d'activació de la xarxa neuronal sigui una funció squashing.

Finalment expliquem com determinar els diferents elements que configuren les xarxes neuronals de propagació cap endavant. Definim la funció cost. Expliquem que minimitzant la funció cost mitcançant el mètode del descent del gradient estocàstic i la taxa d'aprenentatge podem calcular els pesos i biaixos. Al final estudiem diferents funcions d'activació i veiem com afecten a les xarxes neuronals explicant també el vanishing gradient.

2020 Mathematics Subject Classification. 68T07, 68T99

Agraïments

En primer lloc al meu tutor l'Eloi Sans pels ànims, suport i ajuda, sense ell no hagués estat possible.

A la meva família, per permetre-m'ho a aquesta edat.

A les amigues i amics per l'escolta activa, els suggeriments i l'ajuda.

A totes les professores i professors que m'han fet gaudir de les matemàtiques al llarg de la meva vida.

Índex

1 Introducció

'Les xarxes neuronals, o les xarxes neuronals artificials per ser més precís, representen una tecnologia que està arrelada a diferents disciplines: neurociència, matemàtiques , estadística, física, informàtica i enginyeria. Les xarxas neuronals troben aplicacions en diversos camps, com ara modelització, anàlisis de sèries temporals, reconeixement de patrons, processament de senyals... i tenen la virtud d'una important propietat: l'habilitat d'aprendre de les dades d'entrada' (extret de [10]).

1.1 Motivació

El Càlcul Numèric sempre m'ha agradat, quan vaig parlar amb el meu tutor i em va proposar entre d'altres temes el de les xarxes neuronals vaig pensar que seria molt interessant. Sempre havia sentit a parlar de les xarxes neuronals però no tenia clar en que consistien. Seria una bona forma de saber-ho.

Conforme anava llegint sobre el tema, més que les possibles aplicacions que tenien les xarxes neuronals, m'interessava una pregunta que sempre acaba per aparèixer a una persona amb formació matemàtica: per qué funciona?

És per això que m'he centrat en intentar entendre quina matemàtica hi ha darrera del fet que les diferents aplicacions de les xarxes neuronals funcionin.

Aquesta recerca dóna per molt més que un treball d'aquestes carecterístiques, per això m'he centrat en entendre com funcionen les xarxes neuronals de propagació cap endavant, i en estudiar dues vies que ens asseguren que les xarxes neuronals de propagació cap endavant sota certes condicions són aproximadors universals de funcions.

1.2 Estructura de la Memòria

La memòria es divideix en cinc seccions. La primera és la introducció i l'última la conclusió.

A la segona, explico que és una xarxa neuronal de propagació cap endavant de manera resumida, però amb els suficients elements per tenir clara l'estructura. No explico com determinar-ne els diferents components que intervenen en l'estructura, perquè s'expliquen més endavant. Necessitarem aquests conceptes per poder passar a la següent secció.

A la tercera part, exploro dues vies que ens permetran afirmar que sota certes condicions aquestes xarxes són aproximadors universals de funcions.

Per una banda, mitjançant el Teorema de Kolmogorov i la millora d'Sprecher que ens permet poder-lo aplicar a les xarxes neuronals per veure que són aproximadores de funcions.

Per altra banda, una investigació que fa servir Teoria de la Mesura, i que des d'un altre punt de vista arriba a conclusions més fortes.

A la quarta part, acabo d'explicar com funcionen les xarxes neuronals, com 'aprenen' per poder modelitzar les dades que introduïm, la funció cost i, com mitjançant la minimització d'aquesta amb el mètode del descent del gradient, podem configurar els paràmetres que faran que la xarxa funcioni, és a dir, el Backpropagation. També explico la funció d'activació, comentant diferents funcions i com la tria d'aquestes afecta en la xarxa.

2 La Xarxa Neuronal de propagació cap endavant

Una xarxa neuronal és un algoritme que s'utilitza com a eina per aproximar funcions $f : \mathbb{R}^n \to \mathbb{R}^m$.

El nom de xarxa neuronal es fa servir perquè l'estructura recorda a la connexió de les neurones.

Una xarxa neuronal de propagació cap endavant o feedforward (d'ara en endavant direm només xarxa neuronal ja que només parlarem d'aquestes) treballa per capes: cada capa té un nombre de nodes o perceptrons. A cada node hi ha enmagatzemat un nombre. L'algorisme comença amb una capa d'entrada, la capa l-èsima utilitza els nodes de la capa $(l-1)$-èsima per calcular els seus nodes, i finalitza amb una capa de sortida. La profunditat de l'algorisme fa referència al nombre de capes i l'amplada al nombre de nodes per capa, tot i que aquesta amplada pot variar de capa a capa.

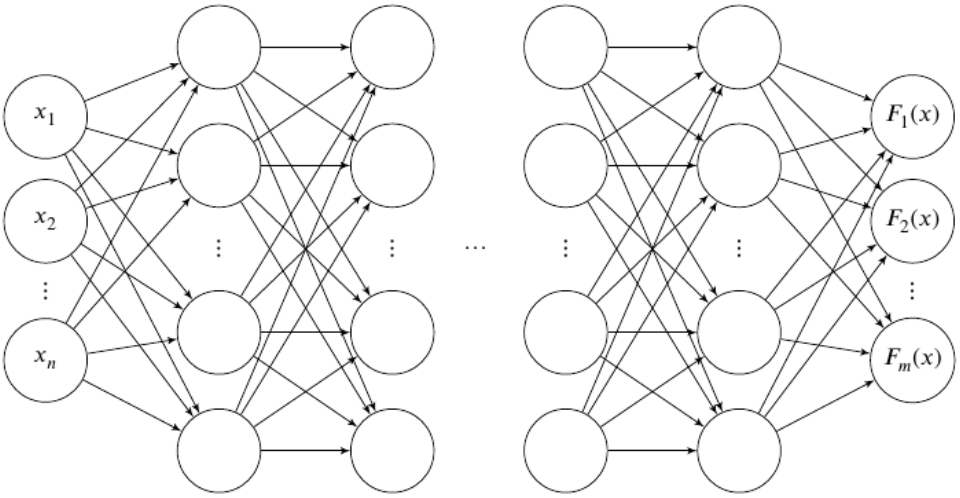


Figura 1: Exemple de xarxa neuronal. Imatge extreta de [3]

L'entrada seran les coordenades d'un punt de $\mathbb{R}^n$ i la sortida haurà de ser el valor aproximat de la funció f en aquest punt. Això ho aconseguirem un cop haguem determinat tots els paràmetres que configuren la xarxa neuronal.

Anem a definir la notació que farem servir per explicitar l'algorisme matemàticament (fem servir la notació de [8] i de [15]):

Definició 2.1. *Notació:*

- $L \in \mathbb{N}$*: serà la profunditat de l'algorisme.*
- $n_l \in \mathbb{N}$ *serà l'amplada de la capa l-èsima.*
- *El vector* $a_i^1 \in \mathbb{R}$ *per* $i = 1, ..., n$ *serà la primera capa de la xarxa neuronal i cada component ocuparà el node que li correspon d'entrada a la xarxa i:*

$$a_i^1 = x_i \qquad i = 1, ..., n_1.$$

En aquest cas $n_1 = n$.

- *El vector $a_i^l \in \mathbb{R}$ per $l = 2, ..., L$ i la i variant segons l'amplada de la capa l, és a dir per $i = 1, ..., n_l$ serà per cada component els nombres que ocupen els nodes de la capa l. En notació vectorial seria:*

$$a_l = \begin{pmatrix} a_1^l \\ \vdots \\ a_{n_l}^l \end{pmatrix}$$

- *Les $w_{i,j}^l \in \mathbb{R}$ seran els pesos entre el node i de la capa $(l-1)$-èsima i el node j de la capa l-èsima, on la $l = 2, ..., L$ i la i i la j recorreran tots els nodes de la capa $(l-1)$-èsima i l-èsima respectivament. Aquests pesos configuren una matriu que serà W_l definida com:*

$$W_l = \begin{pmatrix} w_{1,1}^l & \cdots & w_{1,n_l}^l \\ \vdots & \ddots & \vdots \\ w_{n_{l-1},1}^l & \cdots & w_{n_{l-1},n_l}^l \end{pmatrix}$$

.

- *Les b_j^l seran els biaixos de la capa l per $l = 2, ..., L$ i la j recorrerà tots els nodes de la capa l-èsima. Aquests biaixos de la capa l configuren un vector:*

$$b_l = \begin{pmatrix} b_1^l \\ \vdots \\ b_{n_l}^l \end{pmatrix}$$

- *La funció $\sigma : \mathbb{R} \to \mathbb{R}$ serà la funció d'activació.*

Suposant que la capa $(l-1)$-èsima té k nodes i la capa l-èsima té j nodes tindríem que la fòrmula per calcular la capa l-èsima seria:

$$a_l = \sigma(W_l a_{l-1} + b_l) = \sigma(\begin{pmatrix} w_{1,1}^l & \cdots & w_{1,k}^l \\ \vdots & \ddots & \vdots \\ w_{j,1}^l & \cdots & w_{j,k}^l \end{pmatrix} \begin{pmatrix} a_1^{l-1} \\ \vdots \\ a_k^{l-1} \end{pmatrix} + \begin{pmatrix} b_1^l \\ \vdots \\ b_j^l \end{pmatrix})$$

La funció σ afecta a un vector $v = (v_1, ..., v_n)$ qualsevol de la manera següent:

$$\sigma(v) = \sigma \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} \sigma(v_1) \\ \vdots \\ \sigma(v_n) \end{pmatrix}$$

Finalment la funció que calcularà la xarxa neuronal per aproximar $f : \mathbb{R}^n \to \mathbb{R}^m$, serà una certa funció $F : \mathbb{R}^n \to \mathbb{R}^m$ que vindrà determinada per:

$$F(x) = \sigma(W_L \sigma(...(\sigma(W_2 \sigma(W_1 x + b_1) + b_2)...) + b_L).$$

Si considerem que tot aquest algoritme el volem per poder aproximar una certa funció $f : \mathbb{R}^n \to \mathbb{R}^m$ per poder determinar l'estructura de la xarxa neuronal necessitarem un conjunt $x_1, ..., x_N$ on $x_i \in \mathbb{R}^n$ i $f(x_1), ..., f(x_N)$ on $f(x_i) \in \mathbb{R}^m$ per a $i = 1, ...N$. Aquest conjunt servirà per a que l'algorisme "aprengui", és a dir, per poder determinar el pesos i biaixos que ens permetran poder aproximar la funció en qualsevol punt. Per això utilitzarem la funció cost que serà la diferència entre el valor real de la funció f i l'aproximació que trobem amb l' algorisme. Minimitzant aquesta funció és com podrem trobar els pesos i biaixos. En els següents apartats paralarem de la funció cost i de la forma de minimitzar aquesta funció.

Per altra banda, ens quedaria saber quina és la funció σ i quina amplada i profunditat hauríem de triar per l'algoritme. La tria de la funció σ o funció d'activació la tractem en un proper apartat . Respecte a l'amplada i profunditat de l'algoritme no he trobat cap estudi que doni una regla comuna per triar aquestes variables. A cada camp on s'han aplicat les xarxes neuronals han desenvolupat una recerca per trobar quina és l'amplada i profunditat més idònia per modelitzar els seus problemes.

3 Base teòrica de Xarxes

Hem trobat dues vies per demostrar que amb una xarxa neuronal de tipus feedforward o de propagació cap endavant es pot aproximar una funció $f : \mathbb{R}^n \to \mathbb{R}^m$. Complint per cadascuna de les vies la funció f i la xarxa neuronal certes condicions que detallarem en cada moment.

La primera es basa en el Teorema de Kolmogorov que posteriorment va refinar Sprecher.

La segona es basa en teoria de la mesura.

3.1 Teorema de Kolmogorov

En aquesta secció fem servir els resultats i la notació de [14].

El teorema de Kolmogorov ens dóna una fòrmula per poder expressar una funció de vàries variables com a suma i composició de funcions d'una sola variable.

L'any 1900 Hilbert va enunciar 23 problemes no resolts, la solució dels quals pensava que era important pel desenvolupament de les matemàtiques durant el segle XX. Kolmogorov va enunciar i demostrar aquest teorema al 1957 com a resposta al problema 13 de Hilbert. Temps després es va pensar que aquest teorema podria servir per justificar matemàticament l'estructura de les xarxes neuronals.

A aquesta secció denotarem com a I^n el cub enèsim unitari $[0,1]^n$.

Considerarem els índexs $p, q, k, i \in \mathbb{N}$ tals que:

$$p = 1, ..., n, \quad q = 1, ..., 2n+1, \quad k = 1, 2, ..., \quad i = 1, ..., (9n)^n + 1.$$

Considerem els intervals tancats:

$$A_{k,i}^q = \left[\frac{1}{(9n)^k} \left(i - 1 - \frac{q}{3n} \right), \frac{1}{(9n)^k} \left(i - \frac{1}{3n} - \frac{q}{3n} \right) \right].$$

Veiem les particularitats d'aquesta definició d'intervals.

Comprovem la longitud dels intervals, la distància entre ells i el comportament en modificar els paràmetres.

Veiem quina longitud tenen aquests intervals:

$$\frac{1}{(9n)^k} \left(i - \frac{1}{3n} - \frac{q}{3n} \right) - \frac{1}{(9n)^k} \left(i - 1 - \frac{q}{3n} \right) = \frac{1}{(9n)^k} \left(1 - \frac{1}{3n} \right)$$

Per tant els intervals $A_{k,i}^q$ tenen longitud $\frac{1}{(9n)^k} \left(1 - \frac{1}{3n}\right)$. Notem que aquesta amplada no depèn de i ni de q.

Comprovem quina serà la distància entre els intervals i i $i+1$ si fixem k i q:

$$\frac{1}{(9n)^k} \left((i+1) - 1 - \frac{q}{3n} \right) - \frac{1}{(9n)^k} \left(i - \frac{1}{3n} - \frac{q}{3n} \right) = \frac{1}{3n(9n)^k}$$

Així entre els intervals i i $i+1$ fixades k i q hi ha una distància de $\frac{1}{3n(9n)^k}$.

Comprovem com canvien els intervals si variem q i fixem i i k.

Veiem que si a l'extrem esquerre de l'interval q li restem l'extrem esquerre de l'interval $q+1$:

$$\frac{1}{(9n)^k}\left(i-1-\frac{q}{3n}\right)-\frac{1}{(9n)^k}\left(i-1-\frac{q+1}{3n}\right)=\frac{1}{3n(9n)^k}$$

Com $\frac{1}{3n(9n)^k}>0$ i $\frac{1}{3n(9n)^k}<\frac{1}{(9n)^k}\left(1-\frac{1}{3n}\right)$ l'interval $q=2n+1$ estarà a l'extrem dret i conforme disminueixi q els intervals es desplaçaran a la esquerra i es sobreposaran.

Veiem que si a l'extrem dret de l'interval $q+1$ li restem l'extrem esquerra de l'interval q:

$$\frac{1}{(9n)^k}\left(i-\frac{1}{3n}-\frac{q+1}{3n}\right)-\frac{1}{(9n)^k}\left(i-1-\frac{q}{3n}\right)=\frac{1}{(9n)^k}\left(1-\frac{2}{3n}\right)$$

Com $n\geq 2$ llavors $\frac{1}{(9n)^k}\left(1-\frac{2}{3n}\right)>0$ per tant fixades k i i comprovem de nou que els intervals $q+1$ i q es sobreposen.

Si calculem el valor de l'extrem esquerra del primer interval veiem que:

$$\frac{1}{(9n)^k}\left(1-1-\frac{2n+1}{3n}\right)<0.$$

Per tant els intervals comencen abans del 0.

Si calculem el valor de l'extrem dret de l'últim interval veiem que:

$$\frac{1}{(9n)^k}\left((9n)^k+1-\frac{1}{3n}+\frac{2n+1}{3n}\right)=\frac{1}{(9n)^k}\left((9n)^k+1+\frac{2n}{3n}\right)>1.$$

Per tant podem afirmar:

Lema 3.1. *Els intervals tancats $A^q_{k,i}$ recobreixen $[0,1]$.*

Demostració:: Els càlculs anteriors.

□

Ara definim els cubs:

$$S^q_{k,i_1\ldots i_n}=\prod_{p=1}^{n}A^q_{k,i_p}$$

Si fixem k i q aquests cubs recobriran tot I^n amb una separació de $\frac{1}{3n(9n)^k}$.

A partir d'aqui Kolmogorov enuncia una serie de lemes que no demostra. He intentat la seva demostració però no me n'he sortit.

Aquests lemes són:

Lema 3.2. *:El sistema de cubs $S^q_{k,i_1...i_n}$ amb la k constant i variant la q i les $i_1, ..., i_n$ recobreixen tot el cub I^n de manera que cada punt de I^n està recobert per almenys $n+1$ cubs.*

Lema 3.3. *Les constants $\lambda^{pq}_{k,i}$ i ϵ_k es poden triar de manera que:*

1. $$\lambda^{pq}_{k,i} < \lambda^{pq}_{k,i+1} \leq \lambda^{pq}_{k,i} + 1/2^k.$$

2. $$\lambda^{pq}_{k,i} \leq \lambda^{pq}_{k+1,i'} \leq \lambda^{pq}_{k,i} + \epsilon_k - \epsilon_{k+1}.$$

   si els intervals tancats $A^q_{k,i}$ i $A^q_{k+1,i'}$ no intersecten.

3. *els intervals tancats:*

$$\Delta^q_{k,i_1...i_n} = \left(\sum_{p=1}^{n} \lambda^{pq}_{k,i_p}, \sum_{p=1}^{n} \lambda^{pq}_{k,i_p} + n\epsilon_k \right).$$

   són dos a dos disjunts si fixem k i q.

4. *De 1) i 3) es dedueix que:*

$$\epsilon_k \leq \frac{1}{2^k}.$$

Lema 3.4. *: Si fixem p i q tenim que:*

1. $$\lambda^{pq}_{k,i} \leq \psi^{pq}(x) \leq \lambda^{pq}_{k,i} + \epsilon_k.$$

   és cert per $\forall x \in A^q_{k,i}$ i determina una única funció contínua ψ^{pq} a I^n.

2. $$\sum_{p=1}^{n} \psi^{pq}(x_p) \in \Delta^q_{k,i_1...i_n} \quad per \quad (x_1, ..., x_n) \in S^q_{k,i_1...i_n}.$$

Per construcció ψ^{pq} són monòtones creixents.

Un cop enunciats aquests lemes veiem el Teorema de Kolmogorov on usarem les ψ^{pq} que hem construit.

Teorema 3.5. *: Teorema de Kolmogorov.*

$\forall n \in \mathbb{N}$, $n \geq 2$. Sigui $f(x_1, ..., x_n)$ una funció contínua de I^n a $\mathbb{R}$ qualsevol. Llavors existeixen funcions monòtones creixents ψ^{pq} de I a $\mathbb{R}$ amb $p = 1, ..., n$ i $q = 1, ..., 2n+1$ i χ_q funcions contínues de $\mathbb{R}$ a $\mathbb{R}$ tals que:

$$f(x_1, ..., x_n) = \sum_{q=1}^{2n+1} \chi_q \left[\sum_{p=1}^{n} \psi^{pq}(x_p) \right].$$

Demostració: Per tant pels lemes anteriors ja tenim determinades les funcions ψ^{pq}.

Ara hem de trobar les funcions χ_q.

Les construirem de la forma:

$$\chi^q = \lim_{r\to\infty} \chi_r^q.$$

Abans definim f_r com la funció:

$$f_r(x_1,...,x_n) = \sum_{q=1}^{2n+1} \chi_r^q \left(\sum_{p=1}^{n} \psi^{pq}(x_p) \right).$$

i M_r com:

$$M_r = sup_{x\in I^n} \mid f(x) - f_r(x) \mid .$$

Definirem les χ_r^p de manera recurrent igual que $k_r \in \mathbb{N}$:

$$\chi_0^q \equiv 0.$$

Suposem ara que tenim determinades les funcions contínues χ_{r-1}^q i el nombre k_{r-1}. Per tant f_{r-1} també estarà determinada.

Sigui $\xi_{k,i}^q$ un punt tal que $\xi_{k,i}^q \in A_{k,i}^q$.

Pels intervals tancats $\Delta_{k,i_1..i_n}^q$ definim:

$$\chi_r^q(y) := \chi_{r-1}^q(y) + \frac{1}{n+1}\left(f(\xi_{k,i_1}^q,...,\xi_{k,i_n}^q) - f_{r-1}(\xi_{k,i_1}^q,...,\xi_{k,i_n}^q)\right)$$

Per tant:

$$\mid \chi_r^q(y) - \chi_{r-1}^q(y) \mid \leq \frac{1}{n+1} M_{r-1}.$$

Fora dels intervals tancats $\Delta_{k,i_1..i_n}^q$ definirem la funció χ_r^q de forma arbitrària però preservant la continuitat i l' última desigualtat.

Així tindrem que:

$$f_0 \equiv 0 \quad i \quad M_0 = sup_{x\in I^n} \mid f(x) \mid .$$

Com els diàmetres dels cubs $S_{k,i_1...i_n}^q$ tendeixen a 0 quan $k \to \infty$ podem triar un k_r suficientment gran per a que:

$$\mid f(x) - f_{r-1}(x) \mid < \frac{1}{2n+2} M_{r-1} \quad \forall x \in S_{k_r,i_1...i_n}^q.$$

Així ja tenim definides de forma recurrent tant χ_r^q com k_r.

Ara $\forall (x_1,...,x_n) \in I^n$ tenim que $f - f_r$ la podem escriure com:

$$f(x_1,...,x_n)-f_r(x_1,...,x_n) = f(x_1,...,x_n)-f_{r-1}(x_1,...,x_n)+f_{r-1}(x_1,...,x_n)-f_r(x_1,...,x_n).$$

Reescribint $f_{r-1} - f_r$:

$$f_{r-1}(x_1,...,x_n) - f_r(x_1,...,x_n) = -\sum_{q=1}^{2n+1}\left\{\chi_r^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right) - \chi_{r-1}^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right)\right\}.$$

Per tant tot plegat serà:

$$f(x_1,...,x_n) - f_r(x_1,...,x_n) =$$

$$f(x_1,...,x_n) - f_{r-1}(x_1,...,x_n) + f_{r-1}(x_1,...,x_n) - f_r(x_1,...,x_n) =$$

$$f(x_1,...,x_n) - f_{r-1}(x_1,...,x_n) - \sum_{q=1}^{2n+1}\left\{\chi_r^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right) - \chi_{r-1}^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right)\right\}.$$

Recordem que el punt $(x_1,..,x_n)$ està contingut en $n+1$ cubs $S^q_{k,i_1...i_n}$ Per tant renombrant les q si és necessari podem reescriure el sumatori:

$$\sum_{q=1}^{2n+1}\left\{\chi_r^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right) - \chi_{r-1}^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right)\right\} =$$

$$\sum_{q=1}^{n+1}\left\{\chi_r^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right) - \chi_{r-1}^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right)\right\} +$$

$$+\sum_{q=n+2}^{2n+1}\left\{\chi_r^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right) - \chi_{r-1}^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right)\right\}$$

Ara per cadascun dels termes de la primera part del sumatori tenim:

$$\chi_r^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right) - \chi_{r-1}^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right) =$$

$$= \frac{1}{n+1}\left(f(\xi^q_{k,i_1},...,\xi^q_{k,i_n}) - f_{r-1}(\xi^q_{k,i_1},...,\xi^q_{k,i_n})\right) =$$

$$= \frac{1}{n+1}\left(\frac{1}{2n+2}M_{r-1}\right).$$

Per altra banda a la resta del sumatori tindrem:

$$\sum_{q=n+2}^{2n+1}\left\{\chi_r^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right)-\chi_{r-1}^q\left(\sum_{p=1}^{n}\psi^{pq}(x_p)\right)\right\}$$

$$\leq n\cdot\frac{1}{n+1}M_r-1.$$

Si juntem totes les desigualtats obtingudes tenim que:

$$f(x_1,...,x_n)-f_r(x_1,...,x_n)\leq$$

$$\leq\frac{1}{n+1}\sum_{q=1}^{n+1}\frac{1}{2n+2}M_{r-1}+\frac{n}{n+1}M_r-1=$$

$$=\frac{1}{2n+2}M_{r-1}+\frac{n}{n+1}M_{r-1}$$

$$=\frac{2n+1}{2n+2}M_{r-1}$$

Com que l'última desigualtat és per tot punt $(x_1,...,x_n)\in I^n$ tenim que:

$$M_r\leq\frac{2n+1}{2n+2}M_{r-1}.$$

Per tant:

$$M_r\leq\left(\frac{2n+1}{2n+2}\right)^r M_0.$$

Així si considerem la sèrie:

$$\sum_{r\geq 0}\frac{1}{n+1}M_{r-1}.$$

serà uniformement convergent i com que:

$$\mid\chi_r^q(y)-\chi_{r-1}^q(y)\mid\leq\frac{1}{n+1}M_{r-1}.$$

Per tant la resta és menor que els termes de la sèrie absolutamnet convergent. Així:

$$\chi^q=\lim_{r\to\infty}\chi_r^q.$$

Per tant χ_r^q convergeix uniformement cap a la funció contínua χ^q.

Com que:

$$f_r(x_1,...,x_n) = \sum_{q=1}^{2n+1} \chi_r^q \left(\sum_{p=1}^{n} \psi^{pq}(x_p) \right).$$

i

$$M_r = sup_{x \in I^n} \mid f(x) - f_r(x) \mid .$$

i

$$M_r \leq \left(\frac{2n+1}{2n+2} \right)^r M_0.$$

Si fem $k \to \infty$ ja tenim la prova del Teorema.

□

3.2 Teorema de Sprecher

En aquest apartat fem servir els resultats de [17].

Sprecher va millorar el Teorema de Kolmogorov fent que les $2n^2+n$ funcions ψ^{pq} fossin reemplaçades per una sola ψ més constants i les $2n+1$ funcions χ_q per una sola χ.

Teorema 3.6. *: $\forall N \in \mathbb{N}$, $N \geq 2$ existeix $\psi : [0,1] \to [0,1]$ funció real, monòtona creixent, $\psi(x) \in Lip[\frac{ln2}{ln(2N+2)}]$ depenent de N tal que:*

$\forall \delta \in \mathbb{R}, \delta > 0$, existeix $\epsilon \in \mathbb{Q}, 0 < \epsilon \leq \delta$ i $\forall n \in \mathbb{N}$ i $2 \leq n \leq N$, llavors qualsevol funció contínua $f : I^n \to \mathbb{R}$ es pot representar com:

$$f(x) = \sum_{q=0}^{2n} \chi \left[\sum_{p=1}^{n} \lambda^{pq} \psi(x_p + \epsilon q) \right]$$

on χ és una funció contínua de $\mathbb{R}$ a $\mathbb{R}$ i $\lambda \in \mathbb{R}$ una constant independent de f. Les potències de λ es poden substituir per n nombres reals λ_p per $p = 1,...,n$.

3.3 Teorema d'existència de xarxes neurals de Kolmogorov

En aquesta secció fem servir els resultats de [11].

Aquest teorema utilitza el resultat del Teorema de Sprecher per deduir l'existència de xarxes neuronals de tres capes que poden representar qualsevol funció contínua $f : I^n \to \mathbb{R}^m$.

Teorema 3.7. *:Sigui $f : I^n \to \mathbb{R}^m$, $f(x) = y$, una funció contínua, f pot ser representada exactament per una xarxa neuronal de tres capes tenint a la primera capa les n components del vector $x = (x_1,...,x_n)$, $(2n+1)$ nodes a la segona capa i m nodes a la capa de sortida.*

Els $(2n+1)$ nodes de la segona capa venen determinats per:

$$z_k = \sum_{j=1}^{n} \lambda^k \psi(x_j + \epsilon k) + k \qquad k = 1, ..., 2n+1.$$

on $\lambda \in \mathbb{R}$ *és una constant i* $\psi : [0,1] \to [0,1]$ *és una funció contínua monòtona creixent i* $\psi(x) \in Lip[\frac{ln2}{ln(2N+2}]$, $\epsilon \in \mathbb{Q}$ *tal que* $0 < \epsilon \leq \delta$ *on* $\delta \in \mathbb{R}$ *és una constant positiva arbitrària.*

L'última capa serà:

$$y_i = \sum_{k=1}^{2n+1} g_1(z_k) \qquad i = 1, ..., m.$$

on les funcions g_i *per* $i = 1, ..., m$ *són funcions contínues de* $\mathbb{R}$ *en* $\mathbb{R}$

Per tant $f(x) = y = (y_1, ..., y_m)$.

Demostració: Només cal aplicar el Teorema de Sprecher a cadascuna de les m components del vector $y = f(x)$

□

Aquest teorema ens garanteix que una xarxa neuronal de les característiques que s'-expliciten aproxima exactament qualsevol funció. Ens dóna la certesa que certa xarxa neuronal pot aproximar qualsevol funció. Ara bé, no ens dóna cap algorisme per calcular els diferents elements que conformen la xarxa neuronal.

3.4 Les xarxes neuronals de propagació cap endavant són aproximadors universals

En aquesta secció fem servir els resultat de [13] i també la seva notació.

Definició 3.8. *:$\forall r \in \mathbb{N}$, definim A^r com el conjunt de les funcions afins de $\mathbb{R}^r$ a $\mathbb{R}$, és a dir el conjunt de les funcions de la forma $A(x) = wx + b$ on $w, x \in \mathbb{R}^r$ i $b \in \mathbb{R}$. Per tant:*

$$A^r = \{A(x) | A(x) = wx + b \quad on \quad w, x \in \mathbb{R}^r \quad i \quad b \in \mathbb{R}\}.$$

Al context on ens trobem x correspondrà a l'entrada de la xarxa , els w determinaran els pesos i b determinarà els biaixos de la primera capa.

Definició 3.9. *:Sigui G una funció mesurable de $\mathbb{R}$ a $\mathbb{R}$ i $r \in \mathbb{N}$. Llavors definim $\Sigma^r(G)$ com la classe de funcions:*

$$\Sigma^r(G) = \left\{ \begin{array}{l} f : \mathbb{R}^r \to \mathbb{R} : f(x) = \sum_{j=1}^{q} \beta_j G(A_j(x)), \\ x \in \mathbb{R}^r, \quad \beta_j \in \mathbb{R}, \quad A_j \in A^r, \quad q = 1, 2, ... \end{array} \right\}.$$

Definició 3.10. *: Una funció $\psi : \mathbb{R} \to [0, 1]$ és una funció squashing (traduït literalment seria funció d'aplastament) si no decreix i:*

$$\lim_{x \to +\infty} \psi(x) = 1$$

i:

$$\lim_{x \to -\infty} \psi(x) = 0.$$

Observem que quan G sigui una funció squashing $\Sigma^r(G)$ serà la classe de funcions sortints d'una xarxa neuronal de propagació cap endavant d'una capa oculta amb funcions squashing com a funcions d'activació a la capa oculta i cap funció d'activació a la capa de sortida.

Els escalars β_j determinaran els pesos de la xarxa de la capa oculta a la capa de sortida i en aquest cas la capa de sortida no tindrà cap biaix.

Les funcions squashing pel seu comportament, com a molt, poden tenir un nombre numerable de discontinuitats i per tant són mesurables.

Definició 3.11. *: Per a una funció mesurable G de $\mathbb{R}$ a $\mathbb{R}$ i $\forall r \in \mathbb{N}$ definim $\Sigma\Pi^r(G)$ com la classe de funcions:*

$$\Sigma\Pi^r(G) = \left\{ \begin{array}{l} f : \mathbb{R}^r \to \mathbb{R} : f(x) = \sum_{j=1}^{q} \beta_j \prod_{k=1}^{l_j} G(A_{jk}(x)), \\ x \in \mathbb{R}^r, \quad \beta_j \in \mathbb{R}, \quad A_{jk} \in A^r, \quad l_j \in \mathbb{N}, \quad q = 1, 2, ... \end{array} \right\}.$$

Aquesta $\Sigma\Pi$ xarxa és una xarxa més complicada que la que hem definit nosaltres abans, però no hi ha problema, doncs el que provem per les $\Sigma\Pi$ xarxes també serà cert per les Σ xarxes, ja que les Σ xarxes és un cas particular de les $\Sigma\Pi$ xarxes quan $l_j = 1$ $\forall j$.

Definició 3.12. *: Definim C^r com el conjunt de les funcions contínues de $\mathbb{R}^r$ a $\mathbb{R}$.*

Definim M^r com el conjunt de les funcions Borel-mesurables de $\mathbb{R}^r$ a $\mathbb{R}$.

Definim la σ-àlgebra de $\mathbb{R}^r$ com a B^r. (Això serà important doncs més endavant haurem de definir una mesura de probabilitat sobre aquesta σ-àlgebra).

Si G és Borel-mesurable llavors:

$$\Sigma^r(G) \subset M^r \qquad \Sigma\Pi^r(G) \subset M^r.$$

Si G és contínua llavors:

$$\Sigma^r(G) \subset C^r \qquad \Sigma\Pi^r(G) \subset C^r.$$

També que les classes (ja que sabem per [3] que les funcions contínues son Borel-mesurables):

$$C^r \subset M^r.$$

Les funcions no mesurables són patològiques segons [13].Certes funcions senzilles discontínues a trossos seran mesurables i no contínues. Ens és útil considerar també les funcions mesurables, i no només les contínues, per ampliar el conjunt de funcions que podrem aproximar.

Primer veurem resultats sobre aproximar funcions a C^r, més tard extendrem els resultats sobre aproximar funcions a M^r.

El concepte de proximitat entre dues classes de funcions el descriurem amb el concepte de densitat que ara definirem:

Definició 3.13. *: Sigui (X,ρ) un espai mètric i siguin S i T dos subconjunts d'aquest espai mètric. Direm que S és ρ-dens a T si:*

$$\forall \epsilon \geq 0 \quad i \quad \forall t \in T \quad \exists s \in S \quad tal \quad que \quad \rho(s,t) \leq \epsilon.$$

Direm que S és ρ-dens si:

$$\forall \epsilon \geq 0 \quad i \quad \forall t \in X \quad \exists s \in S \quad tal \quad que \quad \rho(s,t) \leq \epsilon.$$

Als teoremes que veurem a continuació els T o X correspondran a C^r o M^r i S correspondrà a $\Sigma^r(G)$ o a $\Sigma\Pi^r(G)$.

El concepte de proximitat entre dues funcions f i g de C^r o de M^r es mesurarà mitjançant una distància ρ, que definim a continuació.

Definició 3.14. *: Siguin f i g dos funcions, f,g $\in C^r$. Sigui K un compacte K$\subset \mathbb{R}^r$ llavors definim la distància entre f i g sobre K com:*

$$\rho_K(f,g) := \sup_{x \in K} \mid f(x) - g(x) \mid$$

Definició 3.15. *: Un subconjunt S de C^r es diu uniformement dens sobre un compacte K a C^r si:*

$$\forall K \quad compacte \quad K \subset \mathbb{R}^r \quad S \quad és \quad \rho_K - dens \quad a \quad C^r.$$

Definició 3.16. *: Una successió de funcions $\{f_n\}$ direm que convergeix a una funció f uniformement sobre compactes si:*

$$\forall K \subset \mathbb{R}^r \quad \rho_K(f_n, f) \xrightarrow[n\to\infty]{} 0.$$

Abans de passar al primer teorema important sobre aproximació, necessitarem uns quants resultats que ens ajudaran a demostrar-ho. Per això enunciarem el teorema de Stone-Weierstrass i recordarem la definició d'àlgebra que necessitem per tal de poder aplicar el Teorema de Stone-Weierstrass.

Definició 3.17. *: Sigui A una família de funcions reals contínues definides a un conjunt E. A és un àlgebra si es compleixen les condicions següents:*

- *Si f,g $\in$ A llavors f+g $\in$ A.*
- *Si f,g $\in$ A llavors f·g $\in$ A.*
- *Si f $\in$ A i $\lambda \in \mathbb{R}$ llavors λ·f $\in$ A.*

Definició 3.18. *: Sigui A una família de funcions reals contínues. Diem que a separa punts a un cert conjunt E si:*

$$\forall x, y \in E, \quad x \neq y \quad \exists f \in A \quad tal \quad que \quad f(x) \neq f(y).$$

Teorema 3.19. *: Teorema de Stone-Weierstrass:*

Sigui A un àlgebra de funcions contínues reals a un conjunt compacte K. Si A separa punts a K i $\forall$ x $\in$ K, $\exists$ f $\in$ A tal que f(x)$\neq$0, llavors A és ρ_K-densa a l'espai de les funcions reals contínues a K.

Teorema 3.20. *: Sigui G una funció contínua no constant qualsevol de $\mathbb{R}$ a $\mathbb{R}$. Llavors $\Sigma\Pi^r(G)$ és uniformement densa sobre compactes a C^r.*

Demostració: Sigui K $\subset \mathbb{R}^r$ un compacte. Volem aplicar el teorema de Stone-Weierstrass, per tant hem de veure que $\Sigma\Pi^r(G)$ és un àlgebra a K. Si f,g $\in \Sigma\Pi^r(G)$ llavors f i g seran de la forma:

$$f(x) = \sum_{j=1}^{q} \beta_j \prod_{k=1}^{l_j} G(A_{jk}(x)) \quad amb \quad x \in \mathbb{R}^n, \quad \beta_j \in \mathbb{R}, \quad A_{jk} \in A^r, \quad l_j \in \mathbb{N} \quad i \quad q \in \mathbb{N}$$

Per tant si $\lambda \in \mathbb{R}$ tenim que f+g, f $\cdot$ g i $\lambda\cdot$ f també seran de $\Sigma\Pi^r(G)$. Aixi $\Sigma\Pi^r(G)$ és un àlgebra a K.

Veiem ara que $\Sigma\Pi^r(G)$ separa punts a K. Per tant, hem de veure que per:

$$\forall \quad x, y \in K \quad i \quad x \neq y \quad \exists f \in \Sigma\Pi^r(G) \quad tal \quad que \quad f(x) \neq f(y).$$

Podem prendre f(x)=G(A(x)) amb A $\in A^r$ ja que si prenem q=1, β_j=1 i l_j=1 veiem que f(x) $\in \Sigma\Pi^r(G)$. Ara només cal prendre a,b $\in \mathbb{R}$ tals que G(a) $\neq$ G(b) i una A $\in A^r$ tal que A(x)=a i A(y)=b llavors G(A(x)) $\neq$ G(A(y)).

Per veure l'última condició del teorema tornem a prendre f(x)=G(A(x)) i prenem un b $\in \mathbb{R}$ tal que G(b) $\neq 0$ i si fixem:

$$A(x) = 0 \cdot x + b \quad llavors \quad \forall x \in K, \quad G(A(x)) = G(b) \neq 0.$$

Així pel teorema de Stone-Weierstrass tenim que $\Sigma\Pi^r(G)$ és ρ_K-dens a l'espai de les funcions reals contínues a K. Com que ho és per tot K ja tenim demostrat el teorema.

□

Amb aquest últim teorema veiem que les xarxes neuronals de propagació cap endavant $\Sigma\Pi$ són capaces d'aproximar qualsevol funció contínua de $\mathbb{R}^r$ a $\mathbb{R}$ sobre un compacte.

El compacte per tant haurà de contenir tots els valors possibles d'entrada de la xarxa.

La funció d'activació pot ser qualsevol funció contínua no constant.

Per als següents resultats necessitarem una mètrica o una distància que definirem sobre una mesura de probabilitat.

Definició 3.21. *: Sigui μ una mesura de probabilitat a $(\mathbb{R}^r, B^r)$. Si f i g $\in M^r$, direm que són μ-equivalents si:*

$$\mu\{x \in \mathbb{R}^r : f(x) = g(x)\} = 1.$$

Per tal que μ sigui mesura de probabilitat $\mu(\mathbb{R}^r) = 1$.

Les funcions seran μ-equivalents si no coincideixen en un conjunt de probabilitat zero. Així només ens hem d'ocupar de les classes de les funcions equivalents.

La mètrica o distància a les classes de funcions equivalents la definirem així:

Definició 3.22. *: Donada una mesura de probabilitat μ a $(\mathbb{R}^r, B^r)$ i dues funcions f,g $\in M^r$ definim una mètrica o distància:*

$$\rho_\mu : M^r x M^r \to \mathbb{R}^+$$

com:

$$\rho_\mu(f, g) = inf\left\{\epsilon > 0 : \mu\{x :\mid f(x) - g(x) \mid > \epsilon\} < \epsilon\right\}.$$

Veiem com es pot definir de diferents maneres que f_n convergeix cap a f, o dit d'una altre manera:

$$\rho_\mu(f_n, f) \xrightarrow[n\to\infty]{} 0.$$

Mitjançant el següent lema:

Lema 3.23. *:Són equivalents les següents afirmacions:*

1. $\rho_\mu(f_n, f) \xrightarrow[n\to\infty]{} 0.$
2. $\forall \epsilon > 0, \mu\{\mid f_n(x) - f(x) \mid > \epsilon\} \xrightarrow[n\to\infty]{} 0.$
3. $\int min\{\mid f_n(x) - f(x) \mid, 1\}\mu(dx) \xrightarrow[n\to\infty]{} 0.$

Demostració:

- $1 \Leftrightarrow 2$: És inmediat.
- $2 \Rightarrow 3$: Si $\mu\{x :\mid f_n(x) - f(x) \mid > \epsilon/2\} < \epsilon/2$ llavors $\int min\{\mid f_n(x) - f(x) \mid, 1\}\mu(dx) < \epsilon/2 + \epsilon/2 = \epsilon.$
- $3 \Rightarrow 2$: Per la desigualtat de Txebixef i tenint en compte que si d és una distància: $\int d(f_n(x) - f(x))\mu(dx) \xrightarrow[n\to\infty]{} 0.$

□

El lema següent ens relacionarà la convergència uniforme d'una successió de funcions sobre un compacte amb la ρ_μ-convergència.

Lema 3.24. *: Si $\{f_n\}$ és una successió de funcions a M^r que convergeix uniformement sobre compactes a la funció f llavors:*

$$\rho_\mu(f_n, f) \xrightarrow[n\to\infty]{} 0.$$

Demostració: Prenem un $\epsilon > 0$. Pel lema 3.23 és suficient trobar una N $\in \mathbb{N}$ tal que $\forall$ n $\geq$ N tenim que:

$$\int min\{\mid f_n(x) - f(x) \mid, 1\}\mu(dx) < \epsilon.$$

Podem suposar que $\mu(\mathbb{R}^r)$=1. Com $\mathbb{R}^r$ és un espai mètric localment compacte llavors μ és una mesura regular (ho sabem per [9], pàgina 228 Teorema F).

Per tant $\exists$ K compacte tal que K $\subset \mathbb{R}^r$ tal que $\mu(K) > 1 - \epsilon/2$. Prenem un N tal que $\forall n \geq N$ tenim que:

$$sup_{x\in K} \mid f_n(x) - f(x) \mid < \epsilon.$$

Així:

$$\int_{\mathbb{R}^r - K} min\{|\ f_n(x) - f(x)\ |, 1\}\mu(dx) + \int_K min\{|\ f_n(x) - f(x)\ |, 1\}\mu(dx) = \epsilon/2 + \epsilon/2 = \epsilon.$$

□

Ara enunciarem aquest lema que necessitarem per la demostració del proper teorema.

Lema 3.25. *: Per qualsevol mesura finita μ, C^r és ρ_μ-dens a M^r.*

Demostració: Prenem una f $\in M^r$ i un $\epsilon > 0$. Hem de trobar una g $\in C^r$ tal que:

$$\rho_\mu(f, g) < \epsilon.$$

Per un M suficientment gran ,

$$\int min\{|\ f \cdot 1_{\{|f|<M\}}, f\ |, 1\}d\mu < \epsilon$$

Per [9], pàgines 241-242 teorema C i D sabem que $\exists$ una g $\in C^r$ tal que:

$$\int |\ f \cdot 1_{\{|f|<M\}}, g\ |\}d\mu < \epsilon/2$$

Per tant:

$$\int min\{|\ f - g\ |, 1\}d\mu < \epsilon$$

□

Teorema 3.26. *: $\forall$ G funció contínua no constant, $\forall$ $r \in \mathbb{N}$ i una mesura de probabilitat μ a $(\mathbb{R}^r, B^r)$, llavors $\Sigma\Pi^r(G)$ és ρ_μ-dens a M^r.*

Demostració : Donada un funció contínua no constant G pel teorema 3.20 $\Sigma\Pi^r(G)$ és uniformement densa sobre compactes a C^r i com pel lema 3.25 C^r és ρ_μ-densa a M^r per tant $\Sigma\Pi^r(G)$ és ρ_μ-densa a M^r.

□

Per tant el que ens diu aquest últim teorema és que les xarxes de la forma $\Sigma\Pi$ poden aproximar qualsevol funció mesurable. Independentment de la funció G que triem, sempre que sigui contínua i no constant, independentment de la dimensió r de l'espai de sortida i de la mesura de probabilitat μ que triem. En aquest sentit les xarxes $\Sigma\Pi$ són aproximadors universals.

Recordem que les funcions squashing no tenen per què ser contínues, però sabem que tindran com a molt un nombre de discontinuitats numerables. Per tant l'últim teorema no ens garanteix aproximació universal si prenem com a G una funció squashing que no sigui

contínua. En el següent teorema mirarem de solucionar-ho. Abans, però, enunciarem i demostrarem un lema que necessitarem per demostrar el teorema més tard. En aquest lema relacionarem una funció squashing contínua amb un altre funció squashing qualsevol, i per tant incloent-hi les discontínues.

Lema 3.27. *: Sigui F una funció contínua squashing i ψ una funció squashing qualsevol llavors:*

$$\forall \epsilon > 0 \quad \exists H_\epsilon \in \Sigma^1(\psi) \quad tal \quad que \quad sup_{\lambda \in \mathbb{R}} \mid F(\lambda) - H_\epsilon(\lambda) \mid < \epsilon$$

Demostració: Prenem un ϵ tal que $0 < \epsilon < 1$. Hem de trobar unes constants β_j i unes funcions afins A_j amb $j \in \{1, 2, ..., Q-1\}$ amb $Q \in \mathbb{N}$ tals que:

$$sup_{\lambda \in \mathbb{R}} \mid F(\lambda) - \sum_{j=1}^{Q-1} \beta_j \psi(A_j(\lambda) \mid < \epsilon$$

Prenem Q tal que $1/Q < \epsilon/2$.

$\forall j \in \{1, 2, ..., Q-1\}$ prenem $\beta_j = 1/Q$.

Prenem $M > 0$ tal que:

$$\psi(-M) < \epsilon/2Q \quad i \quad \psi(M) > 1.\epsilon/2Q.$$

Com ψ és una funció squashing aquesta M existirà.

$\forall j \in \{1, 2, ..., Q-1\}$ sigui:

$$r_j = sup\{\lambda : F(\lambda) = j/Q\}$$

i:

$$r_Q = sup\{\lambda : F(\lambda) = 1 - 1/2Q\}.$$

Com F és una funció squashing les r_j existiran.

$\forall$ r,s tal que $r < s$ sigui $A_{r,s} \in A^1$ l'única funció afí que satisfà que:

$$A_{r,s}(r) = M \quad i \quad que \quad A_{r,s}(s) = -M$$

Així la aproximació que busquem és:

$$H_\epsilon(\lambda) = \sum_{j=1}^{Q-1} \beta_j \psi(A_{r_j, r_{j+1}}(\lambda)).$$

Així a cada un dels intervals: $(-\infty, r_1], (r_1, r_2], ..., (r_{Q-1}, r_Q], (r_Q, \infty)$ tindrem que

$$\mid F(\lambda) - H_\epsilon(\lambda) \mid < \epsilon.$$

□

Un cop demostrat el lema podem enunciar i demostrar el teorema per qualsevol funció squashing sigui o no contínua.

Teorema 3.28. *:∀ ψ funció squashing i ∀ $r \in \mathbb{N}$ i qualsevol mesura de probabilitat μ a $(\mathbb{R}^r, B^r)$ llavors $\Sigma\Pi^r(\psi)$ és uniformement dens sobre compactes a C^r i ρ_μ-dens a M^r.*

Demostració: Pel Lema 3.24 i el teorema 3.26 és suficient provar que $\Sigma\Pi^r(\psi)$ és uniformement dens sobre compactes a $\Sigma\Pi^r(F)$ per alguna funció contínua squashing F. Per veure això és suficient provar que per qualsevol funció de la forma $\prod_{k=1}^{l} F(A_k(\cdot))$ pot ser uniformement aproximada per elements de $\Sigma\Pi(\psi)$.

Prenem $\epsilon > 0$. Com la multiplicació és una funció contínua i $[0,1]^l$ és compacte llavors $\exists\delta > 0$ tal que si $\mid a_k - b_k \mid < \delta$ per a $0 \leq a_k, b_k \geq 1$ amb $k \in \{1, 2, ..., l\}$

llavors :

$$\mid \prod_{k=1}^{l} a_k - \prod_{k=1}^{l} b_k \mid < \epsilon$$

Pel Lema 3.27 hi ha una funció:

$$H_\delta(\cdot) = \sum_{t=1}^{T} \beta_t \psi(A_t^l(\cdot))$$

Com

$$sup_{\lambda \in \mathbb{R}} \mid F(\lambda) - H_\delta(\lambda) \mid < \delta$$

llavors:

$$sup_{x \in \mathbb{R}} \mid \prod_{k=1}^{l} F(A_k(x)) - \prod_{k=1}^{l} H_\delta A_k(x)) \mid < \epsilon$$

Com $A_t^l(A_k(\cdot)) \in A^r$ hem vist que:

$$\prod_{k=1}^{l} H_\delta(A_k(\cdot)) \in \Sigma\Pi^r(\psi).$$

Per tant $\prod_{k=1}^{l} F(A_{jk}(\cdot))$ pot ser aproximada uniformement per elemnets de $\Sigma\Pi^r(\psi)$.

□

En el següent teorema provarem el mateix peró per xarxes Σ^r, però,per poder fer la demostració d'aquest teorema, necessitarem tres lemas que veurem a continuació.

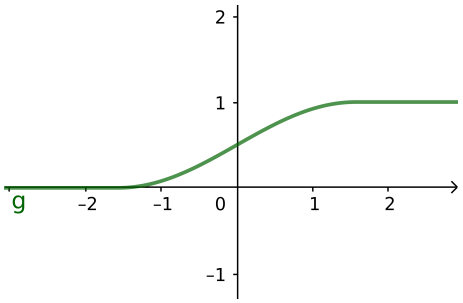
Lema 3.29. *:∀ funció squashing ψ , $\forall\epsilon > 0$ i per $\forall M > 0$ existeix una funció $cos_{M,\epsilon} \in \Sigma^1(\psi)$ tal que:*

$$sup_{\lambda\in[-M,M]} \mid cos_{M,\epsilon}(\lambda) - cos(\lambda) \mid < \epsilon$$

Demostració: Sigui F la funció cosinus de Gallant i White que trobem a [6] definida per:

$$F(x) = \begin{cases} 0 & \text{si } x \leq -\pi/2 \\ (cos(x + 3\pi/2) + 1)/2 & \text{si } -\pi/2 \leq x \leq \pi/2 \\ 1 & \text{si } x \geq \pi/2 \end{cases}$$

La seva gràfica serà:



Veiem que es una funció squashing.

Escalant la funció F podem tenir-la en un interval [-M,M]. El resultat llavors ve d'aplicar el lema 3.27 i la desigualtat triangular.

□

Lema 3.30. *: Sigui:*

$$g(\cdot) = \sum_{j=1}^{Q} \beta_j cos(A_j(\cdot)).$$

Amb $A_j \in A^r$*. Llavors per* $\forall$ *funció squashing* ψ*,* $\forall$ *compacte* $K \subset \mathbb{R}^r$ *i* $\forall \epsilon > 0$*,* $\exists$ $f \in \Sigma^r(\psi)$ *tal que:*

$$sup_{x\in K} \mid g(x) - f(x) \mid < \epsilon.$$

Demostració: Prenem $M > 0$ tal que:

$$\forall j \in \{1, ..., Q\} \quad A_j(K) \subset [-M, M]$$

Com Q és finit , K compacte i $A_j(\cdot)$ és contínua podem trobar aquest M.

Sigui

$$Q' = Q \cdot \sum_{j=1}^{Q} \mid \beta_j \mid .$$

Pel lema 3.29, $\forall x \in K$ tenim:

$$|\sum_{j=1}^{Q} \beta_j cos_{M,\epsilon/Q'}(A_j(x)) - g(x)| < \epsilon$$

Com $cos_{M,\epsilon/Q'} \in \Sigma^1(\psi)$ hem vist que:

$$f(\cdot) = \sum_{j=1}^{Q} cos_{M,\epsilon/Q'}(A_j(\cdot)) \in \Sigma^r(\psi).$$

□

Lema 3.31. *: Per qualsevol funció squashing ψ. $\Sigma^r(\psi)$ és uniformenment densa sobre compactes a C^r.*

Demostració: Pel teorema 3.20 els polinomis trigonomètrics:

$$\left\{ \sum_{j=1}^{Q} \beta_j \prod_{k=1}^{l_j} cos(A_{jk}(\cdot)) \quad tal \quad que \quad Q, l_j \in \mathbb{N}, \beta_j \in \mathbb{R}, A_{jk} \in A^r \right\}.$$

són uniformement densos sobre compactes a C^r.

Aplicant repetidament les identitats trigonomètriques:

$$cos(a) \cdot cos(b) = cos(a+b) - cos(a-b).$$

ens permet reescriure els polinomis trigonomètrics de la forma:

$$\sum_{i=1}^{Q} \alpha_i cos(A_i(\cdot))$$

on $\alpha_i \in \mathbb{R}$ i $A_i \in A^r$. Pel lema 3.30 ja ho tenim.

□

Ara ja podem enunciar el Teorema, tenint en compte els últims tres lemes per la demostració.

Teorema 3.32. *:$\forall$ ψ funció squashing i $\forall$ $r \in \mathbb{N}$ i qualsevol mesura de probabilitat μ a $(\mathbb{R}^r, B^r)$ llavors $\Sigma^r(\psi)$ és uniformement dens sobre compactes a C^r i ρ_μ-densa a M^r.*

Demostració: Pel lema 3.31 sabem que $\Sigma^r(\psi)$ és uniformement dens sobre compactes a C^r. Llavors pel lema 3.24 sabem que $\Sigma^r(\psi)$ es ρ_μ-dens a C^r. Per la desigualtat triangular y el lema 3.25 llavors $\Sigma^r(\psi)$ és ρ_μ-dens a M^r.

□

Per tant del Teorema 3.32 podem dir amb altres paraules que:

- Les xarxes neuronals de propagació cap endavant amb una sola capa oculta poden aproximar qualsevol funció contínua uniformement en qualsevol conjunt compacte, independentment de la funció ψ squashing (en el sentit de si és continua o no), independentment de la dimensió de l'espai de sortida r, i de la mesura μ.
- Les xarxes neuronals de propagació cap endavant amb una sola capa oculta poden aproximar qualsevol funció mesurable amb la mètrica ρ_μ independentment de la funció ψ squashing (en el sentit de si és contínua o no), independentment de la dimensió de l'espai de sortida r, i de la mesura μ.

Per tant en aquest sentit podem dir que les xarxes neuronals Σ són aproximadors universals.

Si ens fixem en les demostracions dels teoremes 3.32 i 3.28 en realitat no necessitem obligatòriament que la funció ψ sigui una funció squashing (contínua o no), sinó que és suficient que la funció ψ sigui tal que les xarxes $\Sigma\Pi^1(\psi)$, en el cas de Teorema 3.28, o $\Sigma^1(\psi)$, en el cas del Teorema 3.32, aproximin una funció squashing en un compacte.

Teorema 3.33. *: Sigui ψ una funció squashing ,$\forall g \in M^r$, $\exists K \subset \mathbb{R}^r$ compacte i una $f \in \Sigma^r(\psi)$ tals que $\forall \epsilon > 0$ tenim que $\mu(K) > 1 - \epsilon$ llavors:*

$$\mid f(x) - g(x) \mid < \epsilon \quad \forall x \in K$$

Independentment de ψ, r o μ.

Demostració: Sigui $\epsilon > 0$. Pel [9] pàgina 243, teorema C i D:

$\exists K^1$ tal que $\mu(K^1) > 1 - \epsilon/2$ i $g \restriction K^1$ (g restringida a K) és contínua a K^1.

Per [5] pàgina 149, Teorema de Tietze:

$\exists g'$ funció contínua $g' \in C^r$ tal que $g' \restriction K^1 = g \restriction K^1$ i:

$$sup_{x \in \mathbb{R}^r} g'(x) = sup_{x \in K^1} g \restriction K^1(x).$$

Pel lema 3.31 $\Sigma^r(\psi)$ és uniformement densa sobre compactes a C^r.

Prenem K^2 un compacte tal que $\mu(K^2) > 1 - \epsilon/2$. Prenem $f \in \Sigma^r(\psi)$ tal que:

$$sup_{x \in K^2} \mid f(x) - g'(x) \mid < \epsilon$$

Llavors:

$$sup_{x \in K^1 \cap K^2} \mid f(x) - g(x) \mid < \epsilon$$

i

$$\mu(K^1 \cap K^2) > 1 - \epsilon$$

□

En aquest últim Teorema el que veiem és que si els valors d'entrada de la funció que volem aproximar estan en un cert compacte K, dins d'aquell compacte és on l'aproximació serà més bona. Conforme la mida μ del compacte sigui més propera a 1, més ben aproximada estarà la funció. El concepte és que conforme més probable sigui que els valors de la funció siguin dins de K, millor serà l'aproximació.

La diferència entre el Teorema 3.33 i el Teorema 3.20 és que la funció g és mesurable en el primer i contínua en el segon i que el compacte K està triat específicament en el primer i és qualsevol al segon.

Ara hem d'introduir el concepte d'espais L_p per continuar amb les següents demostracions.

Definició 3.34. *: $L_p(\mathbb{R}^r, \mu)$ és el conjunt de funcions $f \in M^r$ tals que:*

$$\int \mid f(x) \mid^p \mu(dx) < \infty.$$

La norma als espais L_p serà:

$$\mid\mid f(x) \mid\mid_p = \left[\int \mid f(x) \mid^p \mu(dx)\right]^{1/p}.$$

La mètrica o distància asociada a L_p serà:

$$\rho_p(f, g) = \mid\mid f - g \mid\mid_p .$$

Teorema 3.35. *: Si $\exists K$ compacte, $K \in \mathbb{R}^r$ tal que $\mu(K) = 1$ lllavors $\Sigma(\psi)$ és ρ_μ-dens a $L_p(\mathbb{R}^r, \mu)$ per $\forall p \in [1, \infty)$, independentment de ψ, r o μ.*

Demostració: Prenem $g \in L_p$ i $\epsilon > 0$. Hem de trobar que $\exists f \in \Sigma^r(\psi)$ tal que:

$$\rho_p(f, g) < \epsilon.$$

Dels teoremes 55Ci 55D de [9], pàgines 241-241 sabem que per tota funció acotada $h \in L_p$ existeix una funció contínua f' tal que:

$$\rho_p(h, f') < \epsilon/3.$$

Prenem un M suficientment gran $M \in \mathbb{R}$ i fixem:

$$h = g \cdot 1_{|g| \leq M}$$

Així:

$$\rho_p(g, h) < \epsilon/3$$

Com $\Sigma^r(\psi)$ és uniformement dens sobre compactes, $\exists f \in \Sigma^r(\psi)$ tal que:

$$sup_{x\in K} \mid f(x) - f'(x) \mid^p < (\epsilon/3)^p$$

Com per hipòtesi $\mu(K) = 1$ tenim doncs que:

$$\rho_p(f, f') < \epsilon/3$$

Per tant si ho juntem tot queda:

$$\rho_p(g, f) \leq \rho_p(g, h) + \rho_p(h, f') + \rho_p(f', f) < \epsilon/3 + \epsilon/3 + \epsilon/3 = \epsilon.$$

□

Teorema 3.36. *: Sigui μ una mesura de probabilitat a $[0,1]^r$ llavors $\Sigma^r(\psi)$ és ρ_p-dens a $L_p([0,1]^r, \mu)$ per qualsevol $p \in [1, \infty)$, independentment de ψ, r o μ.*

Demostració: Com $[0,1]^r$ és compacte, només hem d'aplicar el teorema anterior.

□

Teorema 3.37. *: Si μ posa tota la massa 1 a un conjunt finit de punts llavors per cada $g \in M^r$ i $\epsilon > 0$ existeix una $f \in \Sigma^r(\psi)$ tal que:*

$$\mu\{x :\mid f(x) - g(x) \mid < \epsilon\} = 1.$$

Demostració: Sigui $\bar{\epsilon} = min\{\mu(x) : \mu(x) > 0\}$. Llavors per $\forall \epsilon < \bar{\epsilon}$ tenim que:

$$\rho_\mu(f, g) = \epsilon.$$

llavors:

$$\mu\{x :\mid f(x) - g(x) \mid > \epsilon\} = 0.$$

Si apliquem el teorema 3.32 ja estarà.

□

Teorema 3.38. *: Per qualsevol funció Booleana g i cada $\epsilon > 0$ tenim una $f \in \Sigma^r(\psi)$ tal que:*

$$max_{x\in\{0,1\}^r} \mid g(x) - f(x) \mid < \epsilon$$

Demostració: Només cal posar massa $1/2^r$ a cada punt de $\{0,1\}^r$ i aplicar el teorema 3.37.

□

Teorema 3.39. *: Sigui* $\{x_1, ..., x_n\}$ *un conjunt de punts tals que:*

$$\{x_1, ..., x_n\} \quad tal \quad que \quad x_i \in \mathbb{R}^r \quad i \quad x_i \neq x_j \quad si \quad i \neq j.$$

Sigui g *una funció* $g : \mathbb{R}^r \to \mathbb{R}$ *arbitrària. Si* $\exists M \in \mathbb{R} \quad M > 0$ *tals que* $\psi(-M) = 0$ *i* $\psi(M) = 1$ *llavors* $\exists f \in \Sigma^r(\psi)$ *tal que:*

$$f(x_i) = g(x_i) \quad \forall i \in \{1, ..., n\}.$$

Demostració: Farem la demostració en dos pasos, primer ho demostrarem per $\{x_1, ..., x_n\}$ amb $x_i \in \mathbb{R}$ després ho extendrem a $x_i \in \mathbb{R}^r$.

Suposem que $\{x_1, ..., x_n\} \subset \mathbb{R}$ i que, renumerant les x_i si és necesari, $x_1 < x_2 < ... < x_{n-1} < x_n$.

Prenem $M > 0$ tal que $\psi(-M) = 1 - \psi(M) = 0$ i definim:

$$A_1 = M \quad \beta_1 = g(x_1) \quad f^1(x) = \beta_1 \cdot \psi(A_1(x))$$

A_1 serà la funció afí constant $= M$. Per tant:

$$f^1(x) = g(x_1) \quad i \quad per \quad tant \quad f_1(x_1) = g(x_1).$$

Així definim de manera recurrent A_k com:

$$A_{k-1} = -M \quad i \quad A_k = M$$

i

$$\beta_k = g(x_k) - g(x_{k-1}).$$

Per tant si definim:

$$f^k(x) = \sum_{j=1}^{k} \beta_k \psi(A_j(x)).$$

Tenim que:

$$\forall i \leq k \quad f^k(x_i) = g(x_i).$$

Per tant f^n será la funció que busquem.

Pasem ara a $\mathbb{R}^r$. Suposem que $\{x_1, ..., x_n\} \subset \mathbb{R}^r$ amb $r \geq 2$. Prenem un $p \in \mathbb{R}^r$ tal que:

$$Si \quad i \neq j \quad p \cdot (x_i - x_j) \neq 0.$$

Renumerant si és necesari tenim: $p \cdot x_1 < p \cdot x_2 < ... < p \cdot x_{n-1} < p \cdot x_n$. I ara ja som al cas $r = 1$ per tant podem trobar els β_j i els A_j tals que:

$$\sum_{j=1}^{n} \beta_j \psi(A_j(p \cdot x_i)) = g(x_i).$$

Així:

$$f(x) = \sum_{j=1}^{n} \beta_j \psi(A_j(p \cdot x)).$$

Será la funció que buscàvem.

□

Segons [13] modificant aquesta demostració podem provar-ho en el cas que ψ sigui una funció squashing qualsevol, sense necesitat que $\exists M > 0$ tal que $\psi(-M) = 0$ i que $\psi(M) = 1$.

Fins ara a tots els resultats les funcions eren del tipus $f : \mathbb{R}^r \to \mathbb{R}$. Podem donar resultats anàlegs per xarxes de varies sortides, és a dir xarxes neuronals de propagació cap endavant que aproximin funcions contínues o mesurables de $\mathbb{R}^r$ a $\mathbb{R}^s$ amb $r, s \in \mathbb{N}$.

Hauríem de refer les definicions de conjunt de funcions contínues de $\mathbb{R}^r$ a $\mathbb{R}^s$ com a $C^{r,s}$ i el conjunt de funcions mesurables de $\mathbb{R}^r$ a $\mathbb{R}^s$ com a Mr, s.

També hauríem d'extendre el concepte de xarxes de Σ^r o $\Sigma\Pi^r$ a $\Sigma^{r,s}$ i $\Sigma\Pi^{r,s}$ respectivament, reinterpretant les β_j com vectors de components sx1.

La funció $g : \mathbb{R}^r \to \mathbb{R}^s$ tindrà components g_i per $i = 1, 2, ..., s$.

La mètrica o la distància també quedarà redefinida com:

$$\rho_\mu^s = \sum_{i=1}^{s} \rho_\mu(f_i, g_i).$$

La distáncia ρ_p també quedarà redefinida de forma multivariant.

Amb totes aquestes redefinicions podem dir:

Teorema 3.40. *: Els Teoremes 3.28, 3,32, 3.33, 3.35, 3.36, 3.37, 3.38 són certs també per les classes $\Sigma\Pi^{r,s}$ i/o $\Sigma^{r,s}$ aproximant funcions a $C^{r,s}$ i $M^{r,s}$ amb ρ_μ^s i ρ_p.*

Demostració: Usant vectors β_j que siguin 0 menys en la posició i-èsima podem aproximar cada una de les g_i amb una aproximació de ϵ/s. Sumant les s components de les g_i ja tindríem l'aproximació per $\Sigma\Pi^{r,s}$ i $\Sigma^{r,s}$.

□

Així les xarxes neuronals de propagació cap endavant de sortida multiple són aproximadors universals de funcions multivariants.

Fins ara tots els resultats han estat per xarxes neuronals d'una sola capa. El resultat final descriurà les capacitats aproximadores de les xarxes neuronals de propagació cap endavant de sortida multiple i de vàries capes.

Per simplicitat considerarem les xarxes Σ només. Les xarxes $\Sigma\Pi$ com que contenen a les Σ també tindran les mateixes propietats, es podría demostrar també per aquestes xarxes pero segons [13] la complexitat notacional ho faria molt farragós.

Abans d'enunciar el següent teorema necessitarem un lema previ per poder demostrar-lo.

Lema 3.41. *: Sigui F la classe de funcions contínues de $\mathbb{R}$ a $\mathbb{R}$ i G la classe de funcions contínues de $\mathbb{R}^r$ a $\mathbb{R}$ per tant F és uniformement dens sobre compactes a C^1 i G és uniformement dens sobre compactes a C^r.*

La classe $F \circ G$:

$$F \circ G = \{f \circ g : f \in F \quad i \quad g \in G\}$$

és uniformement densa sobre compactes a C^r.

Demostració: Prenem una $h \in C^r$ qualsevol, un $K \subset \mathbb{R}^r$ compacte i un $\epsilon > 0$.

Hem de demostrar que existeix una $f \in F$ i una $g \in G$ tals que:

$$sup_{x \in K} \mid f(g(x)) - h(x) \mid < \epsilon.$$

Per hipòtesi $\exists g \in G$ tal que:

$$sup_{x \in K} \mid g(x) - h(x) \mid < \epsilon/2.$$

Com K és compacte i h és contínua $\{h(x) : x \in K\}$ és compacte. Com $\{g(x) : x \in K\}$ està acotada. Sigui S la clausura compacte de $\{g(x) : x \in K\}$.

Per hipòtesi $\exists f \in F$ tal que:

$$sup_{s \in S} \mid f(s) - s \mid < \epsilon/2.$$

Per tant $f \circ g$ serà la funció que busquem ja que:

$$sup_{x \in K} \mid f(g(x)) - h(x) \mid \leq sup_{x \in K} \mid f(g(x)) - g(x) + g(x) - h(x) \mid \leq$$

$$\leq sup_{x \in K} \mid f(g(x)) - g(x) \mid + sup_{x \in K} \mid g(x) - h(x) \mid < \epsilon/2 + \epsilon/2 = \epsilon.$$

□

Denotarem la classe de les funcions de sortida de les xarxes neuronals de propagació cap endavant de l capes (contant la capa d'entrada peró no la de sortida) que van de $\mathbb{R}^r$ a $\mathbb{R}^s$ usant funcions squashing ψ com $\Sigma_l^{r,s}$. Els casos anteriors eren per $l = 2$.

Les regles d'activació pels elements d'aquesta xarxa seran:

$$a_{ki} = G_k(A_i(a_{k-1})) \quad per \quad i = 1, ..., q_k \quad k = 1, ..., l.$$

on:

- a_k és un vector $q_k x 1$ amb elements a_{ki}
- $a_0 = x$ per convenció (seran les entrades de la xarxa).
- $G_1, ..., G_{l-1} = \psi$ i G_l la identitat.
- $q_0 = r$ i $q_l = s$.

,

Així podem enunciar el següent teorema:

Teorema 3.42. *: El Teorema 3.32, 3.33, 3.35, 3.36, 3.37, 3.38 i 3.40 són valids per les classes $\Sigma_l^{r,s}(\psi)$ aproximant funcions a $C^{r,s}$ i $M^{r,s}$ amb ρ_μ^s i ρ_p multivariant. Amb $l \geq 2$.*

Demostració: Podem considerar el cas $s = 1$ ja que si $s \geq 2$ podem aplicar el Teorema 3.40.

Serà suficient demostrar que $\forall k$ la classe de funcions:

$$J_k = \left[\sum_{i_k} \beta_{i_k} \psi \left(A_{i_k} \left(\sum_{i_k - 1} \beta_{i_k, i_{k-1}} \psi \left(... \left(\sum_{i_1} \beta_{i_k ... i_1} \psi(A_{i_k ... i_1}(x)) \right) ... \right) \right) \right) \right].$$

és uniformement densa sobre compactes a C^r.

Pel lema 3.31 sabem que és cert per $k = 1$. Hem d'aplicar inducció sobre k per completar la prova.

Suposem que J_k és uniformement densa sobre compactes a C^r. Hem de provar que J_{k+1} també ho és, però:

$$J_{k+1} = \left\{ \sum_j \beta_j \psi(A_j(g_j(x)))) : g_j \in J_k \right\}$$

Però el lema 3.31 ens diu que la classe de funcions:

$$\left\{ \sum_j \beta_j \psi(A_j(\cdot))) \right\}.$$

és uniformement densa sobre compactes a C^1.

Pel lema 3.41 tenim doncs que J_{k+1} és uniformement densa sobre compactes a C^r.

Per tant per hipòtesi d'inducció J_k és uniformement densa sobre compactes per $\forall k$.

□

Per tant les $\Sigma_l^{r,s}$ xarxes seran aproximadors universals per funcions multivariants o vectorials.

Segons [13] per tant les xarxes neuronals de propagació cap endavant amb una funció d'activació squashing ψ són capaces d'aproximar qualsevol funció mesurable amb el grau de presició que vulguem.

Això implica que qualsevol error en les aplicacions serà degut a errors en la tria dels pesos i biaixos, és a dir en el procés d'aprenentatge, a un insuficient nombre de capes o bé a que la funció que volem aproximar no te una relació determinista entre les dades d'entrada i de sortida.

4 Base algorítmica de Xarxes

Un cop hem vist la justificació matemàtica de perquè una xarxa neuronal es pot fer servir per aproximar una funció, veiem quins algoritmes fem servir per determinar els paràmetres que configuren la xarxa neuronal. En aquesta secció fem servir resultats de [1],[2],[4],[7],[8],[10],[12] i [15].

4.1 La funció Cost

Si $F : \mathbb{R}^n \to \mathbb{R}^m$ és la funció que realment calcula l'algoritme definirem la funció de cost o funció quadràtica del cost com:

$$C := \frac{1}{2N} \sum_{i=1}^{N} \|f(x_i) - F(x_i)\|_2^2,$$

Si notem

$$C_x := \frac{1}{2N} \|f(x) - F(x)\|_2^2$$

Podríem reescriure C com:

$$C = \sum_{i=1}^{N} C_{x_i}.$$

La funció cost així definida ens calcula l'error d'aproximació de f per F. Per tant si volem que F aproximi bé f hem de minimitzar C. Finalment ens trobem amb un problema numèric de calcular mínims i mitjançant aquests càlculs trobar els millors pesos i biaixos. Per minimitzar C podem fer servir el mètode del descent del gradient i el Backpropagation, tot i que hi ha d'altres mètodes per minimitzar la funció cost.

4.2 Descent del gradient

Veiem com funciona en general el mètode del descent del gradient per calcular mínims d'una funció $g : \mathbb{R}^n \to \mathbb{R}$. Fem servir els resultats i notació de [4].

Definició 4.1. *Sigui $g : \mathbb{R}^n \to \mathbb{R}$ una funció diferenciable. El gradient de g en $x = (x_1, x_2, ..., x_n)^t$ es denota com $\nabla g(x)$ i es defineix com:*

$$\nabla g(x) = \left(\frac{\partial g}{\partial x_1}(x), ..., \frac{\partial g}{\partial x_n}(x) \right).$$

Sabem que la funció g pot tenir un mínim relatiu en x només quan el gradient sigui zero.

També sabem que si $v = (v_1, ...v_n)^t$ és un vector unitari de $\mathbb{R}^n$, es a dir $\|v\|_2^2 = 1$, la derivada direccional de g en x en la direcció v és:

$$D_v g(x) = \lim_{h \to 0} (\frac{1}{h}(g(x + hv) - g(x)) = v^t \nabla g(x).$$

Per tant la derivada direccional mesura el canvi de valor de la funció g respecte al canvi de la variable en la direcció v i la direcció que estableix el màxim valor de la derivada direccional serà $\nabla g(x)$ sempre que $\nabla g(x) \neq 0$. Així, si volem trobar la direcció de la màxima disminució de g en x serà $-\nabla g(x)$.

L'algorisme del descent del gradient per calcular el mínim d'una funció $g : \mathbb{R}^n \to \mathbb{R}$, serà un procés iteratiu. Donat $x_0 \in \mathbb{R}^n$ i una $\lambda \in \mathbb{R}, \lambda > 0$ definirem el següent iterat com:

$$x_{i+1} := x_i - \lambda \nabla g(x_i). \qquad i = 1, 2, ...$$

A cada pas ens hem d'assegurar que:

$$g(x_{i+1}) < g(x_i).$$

I pararem quan:

$$|g(x_{i+1}) - g(x_i)| < TOL$$

Si definim TOL com la tolerància mínima que volem per a aproximar el mínim.

La funció cost abans definida és de la forma $C : \mathbb{R}^n \to \mathbb{R}^+$. Ès a dir que sempre pren valors positius, això ens dóna alguns aventatges.

El que volem en cas de les xarxes neuronals és que la funció cost sigui zero per tal que no hi hagi diferència entre la funció que volem aproximar i el resultat que ens dóna la xarxa neuronal. Aquest zero serà el mínim que volem trobar mitjançant el descent del gradient, i allà on el gradient de la funció cost valdrà zero. Per tant incrementarem la funció cost variant el gradient mitjançant un vector v:

$$\triangle C \approx \nabla C \triangle v.$$

On v serà, fent servir el mètode del descent del gradient:

$$\triangle v = -\lambda \nabla C$$

Per tant:

$$\triangle C \approx -\lambda \nabla C \nabla C = -\lambda \|\nabla C\|^2$$

Així ens assegurem que com:

$$\|\nabla C\|^2 \geq 0$$

l'increment de C serà sempre negatiu, i per tant disminuirà. El que no sabem amb seguretat és que aquest mínim serà zero.

A λ li diem taxa d'aprenentatge. La tria d'aquesta taxa pot ser molt important, ja que pot variar la velocitat de convergència, pot fer que no convergeixi o que convergeixi a un mínim local i no global.

Segons [15] la taxa d'aprenantge es va variant manualment per tal de trobar la millor aproximació i d'aquesta manera evitar els errors de convergència que poguem trobar.

4.3 Backpropagation

Amb el Backpropagation determinarem els pesos i biaixos de la xarxa neuronal de propagació cap endavant fent servir la funció cost abans definida i el mètode del descens del gradient.

Observem que la funció cost tal i com l'hem definida sembla que depengui de x, però en realitat no és així.

Quan fem la propagació cap endavant tenim uns valors d'entrada i uns valors de sortida donats per les dades observades que volem modelitzar a traves de la xarxa neuronal, aquestes dades no les podem moure, son les dades que volem aproximar.

Per tant quan fem la diferència entre els valors de sortida de la xarxa neuronal i els valors que volem aproximar, aquesta diferència, la funció cost que volem minimitzar, no depèn de x sinó dels pesos w i biaixos b que volem trobar per a què la xarxa realment aproximi la funció que busquem.

Així el descent del gradient l'hem d'aplicar sobre els pesos i biaixos de la xarxa neuronal que són realment les variables de les que depèn la funció cost que volem minimitzar.

El repte, doncs, que hem d'enfrontar ara és el càlcul de $\frac{\partial C}{\partial w^l_{i,j}}$ i de $\frac{\partial C}{\partial b^l_j}$ per tal de poder aplicar el mètode del descent del gradient per minimitzar la funció cost. Això és el Backpropagation o propagació cap enrera. Rep aquest nom ja que comencem amb les diferències que obtenim a la última capa de la xarxa neuronal o capa de sortida i anem calculant cap enrera les derivades parcials de cada capa fins arribar a la primera.

Reescribint el vector a^l tenim:

$$a^l_j = \sigma(\sum_{i=1}^{k} w^l_{j,i} a^{l-1}_i + b^l_j)$$

Si definim z^l_j com:

$$z^l_j := \sum_{i=1}^{k} w^l_{j,i} a^{l-1}_i + b^l_j$$

llavors

$$a^l_j = \sigma(z^l_j).$$

Per tant per calcular $\frac{\partial C}{\partial w^l_{i,j}}$ i $\frac{\partial C}{\partial b^l_j}$ fent servir la regla de la cadena tenim:

$$\frac{\partial C}{\partial w^l_{i,j}} = \frac{\partial C}{\partial z^l_j}\frac{\partial z^l_j}{\partial w^l_{i,j}} = \frac{\partial C}{\partial z^l_j} a^{l-1}_i$$

$$\frac{\partial C}{\partial b_j^l} = \frac{\partial C}{\partial z_j^l}\frac{\partial z_j^l}{\partial b_j^l} = \frac{\partial C}{\partial z_j^l}$$

Ja que

$$\frac{\partial z_j^l}{\partial w_{i,j}^l} = a_i^{l-1}$$

$$\frac{\partial z_j^l}{\partial b_j^l} = 1$$

Suposant que la xarxa neuronal té L capes, fent servir de nou la regla de la cadena i que $a_j^L = \sigma(z_j^L)$ tenim que a l'última capa:

$$\frac{\partial C}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L}\frac{\partial a_j^L}{\partial z_j^l} = \frac{\partial C}{\partial a_j^L}\sigma'(z_j^L)$$

Usant la definició de funció cost i si $F(x) = A_L = (a_1^L, ..., a_k^L)$ és la funció que calcula la xarxa neuronal i $f(x) = (y_1, ..., y_k)$ es la funció que volem modelitzar, tenim:

$$\frac{\partial C}{\partial z_j^L} = (a_j^L - y_j)\sigma'(z_j^L)$$

Amb això ja podem aplicar el descent del gradient a l'última capa de la xarxa neuronal canviant les $w_{i,j}^L$ i les b_j^L per unes altres que calculin millor l'aproximació de $f(x)$.

A partir de l'última capa podem anar calculant les anteriors. Si ja tenim calculades les $w_{i,j}^l$ i les b_j^l podem trobar les $w_{i,j}^{l-1}$ i les b_j^{l-1}, així:

$$\frac{\partial C}{\partial z_j^{l-1}} = \sum_{i=1}^{k} \frac{\partial C}{\partial z_i^l}\frac{\partial z_i^l}{\partial z_j^{l-1}}.$$

Centrant-nos ara en $\frac{\partial z_i^l}{\partial z_j^{l-1}}$ tenim:

$$\begin{aligned}
\frac{\partial z_i^l}{\partial z_j^{l-1}} &= \frac{\partial(\sum_{p=1}^{k} w_{i,p}^l a_p^{l-1} + b_i^l)}{\partial z_j^{l-1}} \\
&= \frac{\partial(\sum_{p=1}^{k} w_{i,p}^l \sigma(z_p^{l-1}) + b_i^l)}{\partial z_j^{l-1}} \\
&= \frac{\partial(w_{i,j}^l \sigma(z_j^{l-1}))}{\partial z_j^{l-1}} \\
&= w_{i,j}^l \sigma'(z_j^{l-1}).
\end{aligned}$$

Per tant:

$$\frac{\partial C}{\partial z_j^{l-1}} = \sum_{i=1}^{k} \frac{\partial C}{\partial z_i^l} w_{i,j}^l \sigma'(z_j^{l-1}).$$

Recuperant de les anteriors equacions de $\frac{\partial C}{\partial w_{i,j}^l}$ i de $\frac{\partial C}{\partial b_j^l}$ tenim:

$$\frac{\partial C}{\partial w_{i,j}^{l-1}} = \frac{\partial C}{\partial z_j^{l-1}} a_i^{l-2} = a_i^{l-2} \sum_{i=1}^{k} \frac{\partial C}{\partial z_i^l} w_{i,j}^l \sigma'(z_j^{l-1}).$$

$$\frac{\partial C}{\partial b_j^{l-1}} = \frac{\partial C}{\partial z_j^{l-1}} = \sum_{i=1}^{k} \frac{\partial C}{\partial z_i^l} w_{i,j}^l \sigma'(z_j^{l-1}).$$

Per tant ja podem explicitar l'algoritme que farà que la xarxa neuronal aprengui. Abans , però, hem de decidir com inicialitzar els pesos i biaixos que farem servir en el càlcul de la primera aproximació. Segons [15] es fa servir una distribució normal: $N(0,1)$ per inicialitzar tots aquests paràmetres.

Així l'algoritme del backpropagation o propagació cap enrera amb el descent del gradient quedaria així:

- Entrada: El valor $L \in \mathbb{N}$ que serà la profunditat de la xarxa o el nombre de capes de la xarxa. Per $l = 2, ..., L$ els pesos $w_{i,j}^l \in \mathbb{R}$ i biaixos $b_j^l \in \mathbb{R}$ inicials, la i i la j vindrà determinada per l'amplada que tingui cada capa de la xarxa neuronal. El valor $x \in \mathbb{R}^n$ i $y = f(x) \in \mathbb{R}^m$ de la funció que volem modelitzar. El valor $\lambda \in \mathbb{R}$ o taxa d'aprenentatge. $TOL \in \mathbb{R}$ la tolerància que serà un valor molt petit a partir del qual deixarem de fer backpropagation i $N \in \mathbb{N}$ com a nombre màxim d'iteracions del backpropagation.
- Sortida: Els nous pesos $w_{i,j}^l$ i biaixos b^l que aproximen la funció o un missatge que s'ha superat el nombre màxim d' iteracions del backpropagation N.
- Pas 1: El valor del vector $x \in \mathbb{R}^n$ l'igualem a a^1
- Pas 2:

$$iteració = 1.$$

- Pas 3: Mentre $iteracio < N$ fer els passos de 4 a 10 .
- Pas 4: Per $l = 2, 3, .., L$.

$$z^l = w^l a^{l-1} + b^l$$

$$a^l = \sigma(z^l)$$

- Pas 5: Per $j = 1, ..., m$

$$\delta_j^L = (a_j^L - y_j)$$

- Pas 6: Si per $j = 1, ..., m$

$$\delta_j^L < TOL\ PAREM.$$

Sortida: Els pesos $w_{i,j}^l$ i biaixos b^l.

- Pas 7:

$$\delta_j^L = (a_j^L - y_j)\sigma'(z_j^L)$$

- Pas 8: Per $l = L-1, L-2, ..., 2$

$$\delta_j^l = \sum_{i=1}^{k} \delta_i^{l+1} w_{i,j}^{l+1} \sigma'(z_j^l)$$

- Pas 9: Per $l = L, L-1, ..., 2$

$$w_{i,j}^l = w_{i,j}^l - \lambda a_i^{l-1} \delta_j^l$$

$$b_j^L = b_j^L - \lambda \delta_j^l$$

- Pas 10:

$$iteració = iteració + 1$$

- Pas 11: Sortida: S'ha excedit el nombre màxim d'iteracions.

Per tant ja tenim l'algoritme per a què una xarxa neuronal "aprengui", és a dir, per poder triar el pesos i biaixos que millor aproximin una funció de la qual tenim una entrada i una sortida. Però habitualment les xarxes es fan servir quan tenim un conjunt considerable de dades d'entrada i sortida. Per això es fa servir el descent del gradient estocàstic.

4.4 Descent del gradient estocàstic

Per 'entrenar' una xarxa neuronal tindrem un conjunt de dades $x_1, ..., x_N$ i $f(x_1) = y_1, ..., f(x_N) = y_N$ on $x_i \in \mathbb{R}^n$ i $y_i \in \mathbb{R}^m$ per $i = 1, ..., N$, i normalment aquesta N serà molt gran. Com:

$$\nabla C = \frac{1}{N} \sum_{i=1}^{N} \nabla C_{x_i}.$$

El conjunt de càlculs per arribar a trobar ∇C serà molt gran.

És per això que es fa servir el descent del gradient estocàstic. Amb el descent del gradient estocàstic triem un conjunt més petit de dades $x_1, ..., x_{\hat{N}}$ i aproximem:

$$\nabla C \approx \frac{1}{\hat{N}} \sum_{i=1}^{\hat{N}} \nabla C_{x_i}.$$

Així canviarem els pesos:

$$w_k = w_k - \frac{\lambda}{\hat{N}} \sum_{i=1}^{\hat{N}} \frac{\partial C_{x_i}}{\partial w_k}$$

I els biaixos per:

$$b_l = b_l - \frac{\lambda}{\hat{N}} \sum_{i=1}^{\hat{N}} \frac{\partial C_{x_i}}{\partial b_l}$$

Un cop recalculats els pesos i biaixos es prova la xarxa neuronal amb les dades restants per comprovar si aproxima bé la resta de dades. Si no fos així es recalcularien pesos i biaixos fins que la xarxa neuronal aproximi tot el conjunt de dades.

A la tria d'aquestes dades amb la intenció que aproximin la resta del conjunt de dades totals s'anomena entrenament supervisat.

Aquesta tria de dades en dos grups, un per entrenar la xarxa i l'altre per comprovar els resultats de l'entrenament per veure si ha estat efectiu, és força important per arribar a aproximar correctament la funció desitjada.

4.5 La funció d'activació

La precisió en l'aproximació de la xarxa neuronal també vindrà definida pel tipus de funció d'activació que fem servir.

Si la funció d'activació no existís en la xarxa neuronal, funcionaria com una mena de regressió lineal ja que com hem vist:

$$a^l = \begin{pmatrix} w^l_{1,1} & \cdots & w^l_{1,k} \\ \vdots & \ddots & \vdots \\ w^l_{j,1} & \cdots & w^l_{j,k} \end{pmatrix} \begin{pmatrix} a^{l-1}_1 \\ \vdots \\ a^{l-1}_k \end{pmatrix} + \begin{pmatrix} b^l_1 \\ \vdots \\ b^l_j \end{pmatrix}$$

Per altra banda si fem servir una funció d'activació que sigui lineal les dades d'entrada i de sortida seran força similars ja que només les variarem d'una manera lineal. Per tant la xarxa només s'adaptarà a canvis lineals sobre les dades d'entrada. És per això que habitualment es fan servir funcions d'activació no lineals.

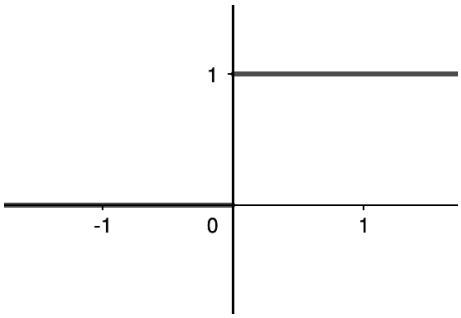
Com que volem utilitzar les xarxes neuronals com una eina d'aproximadors universals de funcions de qualsevol tipus haurem de fer servir la majoria de vegades funcions d'activació no lineals. L'excepció podria ser quan sabem que la funció a aproximar és semblant a una lineal, però llavors hi ha mètodes menys costosos en termes de càlculs i programació per poder-la aproximar, com ara la regressió lineal.

Una altre qüestió important que hem vist sobre la funció d'activació és que ha de ser diferenciable doncs, com hem vist a l' hora d'aplicar el Backpropagation, hem de fer servir la seva derivada.

Per tan la funció d'activació serà: $\sigma : \mathbb{R} \to \mathbb{R}$ diferenciable, com a mínim en un sentit numèric, és a dir que podem donar-li un valor a $\sigma'(z) \forall z \in \mathbb{R}$.

Veiem algunes de les diferents funcions d'activació que es fan servir:

- Funció binària:



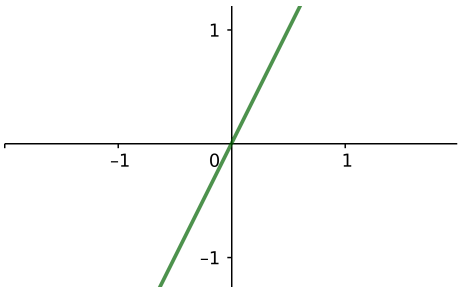
$$\sigma(x) = \begin{cases} 0 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

Aquesta funció es fa servir quan volem que un node s'activi o no, segons el valor que li arribi d'entrada. Amb això aconseguim que aquest node no tingui sortida cap a la següent capa de la xarxa neuronal si el valor d'entrada supera una certa cota.

La funció binària és la més simple de les funcions d'activació. Generalment s'usa per crear un classificador binari, però no es fa servir per un classificador de vàries opcions.

La derivada de la funció binària la considerem zero a efectes numèrics. Amb aquesta derivada no podem fer servir el Backpropagation, per tant no podem canviar mitjançant el descent del gradient els pesos i biaixos. S'hauria de fer de manera manual, per tan seria un funció per problemes de classificació poc complexos.

- Funció lineal:



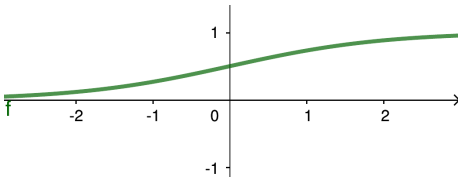
$$\sigma(x) = ax$$

La sortida de la funció d'activació lineal serà directament proporcional a l'entrada.

Aquí la derivada de la funció serà un valor constant que serà independent del valor d'entrada de la funció d'activació. Per tant els canvis que farem durant el Backpropagation seran sempre els mateixos.

Els canvis que farà a la xarxa neuronal aquesta funció d'activació no seran molt complexos, per tant es fa servir per aproximar funcions semblants a les lineals.

- Funció sigmoide:



$$\sigma(x) = \frac{1}{1+e^{-x}} =: sigmoid(x)$$

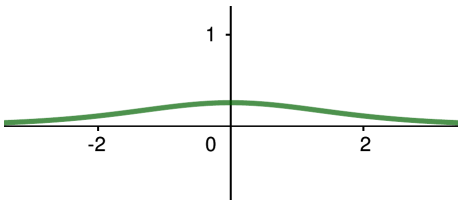
Aquesta és una de les funcions d'activació més usada. La definim com $sigmoid(x)$ perquè la farem servir més endavant per reescriure algunes de les altres funcions d'activació en funció de la sigmoide.

La funció sigmoide transforma els valors d'entrada en valors de sortida entre 0 i 1. Veiem que:

$$\lim_{x \to \pm\infty}(\sigma(x)) = \begin{cases} 1 & \text{si } x \to +\infty \\ 0 & \text{si } x \to -\infty \end{cases}$$

Per tant per molt gran que sigui un valor no canviarà molt la sortida cap a la següent capa, per contra els valors molt grans negatius no afectaran gens la següent capa. També veiem que tots els valors de sortida de la funció seran positius.

La funció sigmoide és diferenciable i la seva derivada és:



$$\sigma'(x) = \frac{e^{-x}}{(1+e^{-x})^2}$$

Podem veure :

$$\lim_{x \to \pm\infty}(\sigma'(x)) = 0$$

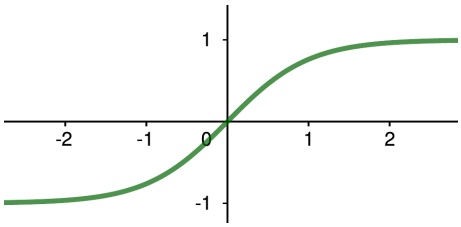
Per tant quan els valors de z^l es fagin molt grans en valor absolut la derivada de la funcio sigmoide retornará un valor molt proper a zero. Això farà que al Backpropagation els canvis en els pesos i biaixos siguin molt petits, per tant la xarxa aprendrà molt a poc a poc. Això es coneix com el problema del vanishing gradient o esvaniment del gradient.

La derivada de la funció sigmoide la podem reescriure en funció d'ella mateixa com:

$$\sigma'(x) = \sigma(x)(1 - \sigma(x))$$

Aquesta propietat de la derivada ens farà més fàcil la programació del Backpropagation en el sentit del nombre d'operacions a fer.

- Funció tangent hiperbòlica:



$$\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{2}{1 + e^{-2x}} - 1$$

Aquesta funció és molt semblant a la sigmoide, de fet:

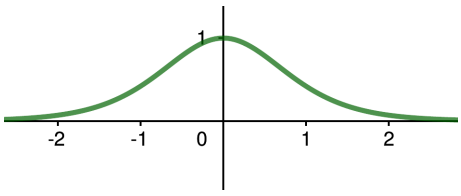
$$\sigma(x) = 2sigmoid(2x) - 1$$

Però transforma els valors d'entrada en valors de sortida entre -1 i 1, això és un canvi respecte la funció sigmoide ja que el signe de sortida pot variar entre positiu i negatiu. També veiem que:

$$\lim_{x \to \pm\infty} (\sigma(x)) = \begin{cases} 1 & \text{si } x \to +\infty \\ -1 & \text{si } x \to -\infty \end{cases}$$

Per tant, com a la funció sigmoide, els valors molt grans no afectaran molt a la sortida de la funció, mentre que els valors molt grans negatius simplement canviran el signe de l'entrada.

La derivada d'aquesta funció d'activació és:



$$\sigma'(x) = 1 - (\frac{e^x - e^{-x}}{e^x + e^{-x}})^2$$

Podem veure :

$$\lim_{x \to \pm\infty} (\sigma'(x)) = 0$$

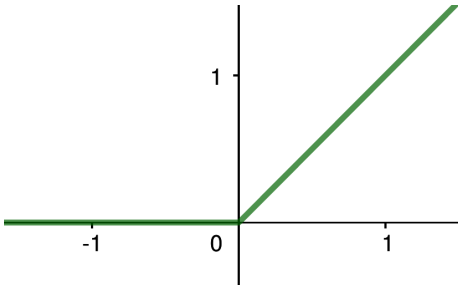
Per tant com ja hem explicat a l'apartat de la funció sigmoide tenim el problema del vanishing gradient o esvaniment del gradient.

La derivada de la funció tangent hiperbòlica la podem reescriure en funció d' ella mateixa com:

$$\sigma'(x) = 1 - (\sigma(x))^2$$

Aquesta propietat de la derivada ens farà més fàcil la programació del Backpropagation en el sentit del nombre d'operacions a fer.

- Funció ReLU:



$$\sigma(x) = \begin{cases} 0 & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases}$$

La funció ReLU rectifica la funció lineal i juntament amb les seves variants és bastant usada a xarxes neuronals. El nom ReLU ve d'unitat lineal rectificada per les seves sigles en anglès.

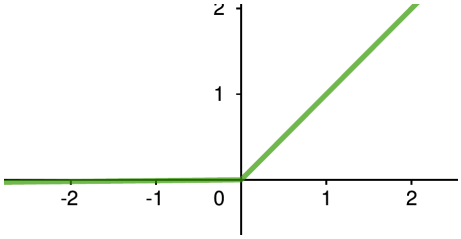
Aquesta funció d'activació fa que quan la sortida de $z^l < 0$ la neurona no s'activi, és a dir que no afectarà a la següent capa de la xarxa neuronal.

La derivada numèrica de la funció ReLU la considerem de la següent manera:

$$\sigma'(x) = \begin{cases} 0 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

En els casos en el que la derivada sigui zero durant el Backpropagation els pesos i biaixos d'aquest nodes no canviaran.

- Funció Leaky ReLU:



$$\sigma(x) = \begin{cases} 0.01x & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases}$$

Amb aquesta variant i les properes que descrivim de la funció ReLU s'aconsegueix que quan:

$$z^l = w^l a^{l-1} + b^l < 0$$

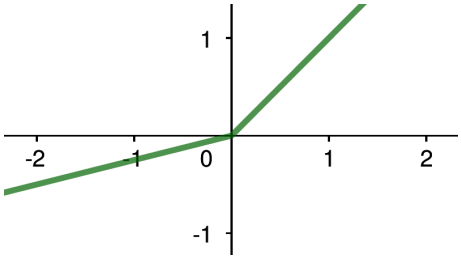
la neurona s'activi però afecti poc a la següent capa neuronal.

La derivada numèrica de la funció Leaky ReLU la considerem de la següent manera:

$$\sigma'(x) = \begin{cases} 0.01 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

Veiem que a diferència de la funció ReLU la derivada no és zero quan $z^l < 0$ i per tant el Backpropagation funciona afectant a més neurones que amb la funcio ReLU.

- Funció ReLU parametritzada:



$$\sigma(x) = \begin{cases} ax & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases}$$

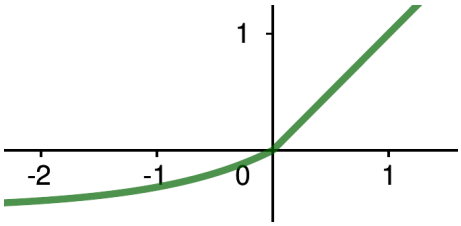
Aquesta funció és semblant a la Leaky ReLU però segons [2] es pot entrenar també el valor a de la funció.

La derivada numèrica de la funció Leaky ReLU la considerem de la següent manera:

$$\sigma'(x) = \begin{cases} a & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

Per tant el comportament a el Backpropagation serà similar al de la Leaky ReLU.

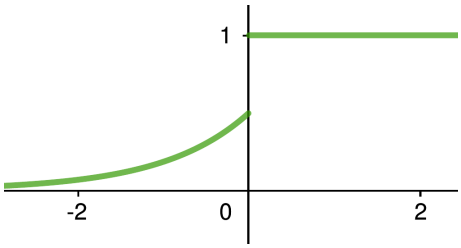
- Funció ELU:



$$\sigma(x) = \begin{cases} a(e^x - 1) & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases}$$

Aquesta funció d'activació es diu funció ELU o funció unitat lineal exponencial i és una altra variació de la funció ReLU. Com a variant de la funció ReLU quan $z^l < 0$ afectarà poc a la següent capa però menys conforme z^l s'apropi molt a zero i més conforme s'alluny del zero.

La derivada numèrica de la funció ELU la considerem de la següent manera:



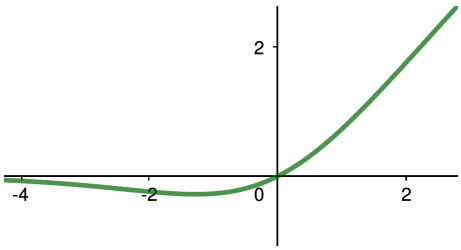
$$\sigma'(x) = \begin{cases} ae^x & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

Aqui de nou com:

$$\lim_{x \to -\infty} (\sigma'(x)) = 0$$

Podem tenir que per valors d'entrada molt grans negatius el Backpropagation funcioni molt lentament.

- Funció Swish:



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

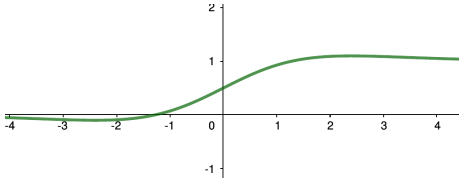
Que podem reescriure com:

$$\sigma(x) = xsigmoid(x)$$

Aquesta funció d'activació va ser descoberta per investigadors de GOOGLE. És una altre variació de la ReLU i la característica principal és que no es monòtona creixent, per tant els valors de sortida de la funció poden decréixer encara que els valors d'entrada creixin.

En ser un funció no creixent cap de les vies per demostrar que les xarxes neuronals de propagació cap endavant són aproximadors universals no ens serveixen per aquesta funció d'activació.

La derivada de la funció Swish és:



$$\sigma'(x) = \frac{xe^{-x} + e^{-x} + 1}{(1 + e^{-x})^2}$$

La diferència principal de la derivada de la funció Swish respecte a les de les altres variants de la funció ReLU és que aquesta si és contínua. Això farà que els canvis al Backpropagation que es produeixin de valors d'entrada al voltant del zero siguin continus.

Reescribint la derivada respecte la funció sigmoide:

$$\sigma'(x) = (1 + x - xsigmoid(x))sigmoid(x)$$

La tria de la funció d'activació més adequada és molt important però no hi ha una regla general per aquesta elecció.

Hem de tenir en compte també que la funció d'activació pot no ser la mateixa en tota la xarxa neuronal. Hi ha molts casos on a les capes interiors de la xarxa es fa servir una funció d'activació i a la capa final o capa de sortida es fa servir una diferent. Per exemple, les funcions ReLU es fan servir a les capes interiors de les xarxes neuronals i no a la capa de sortida.

A cada camp on s'han aplicat les xarxes neuronals s'ha fet investigació per tal de trobar la funció d'activació més idònia per a què la xarxa modelitzi més fidelment la funció a aproximar, i també per a què el nombre d'iterats a fer amb el backpropagation o procés d'aprenentatge per trobar els pesos i biaixos més convenients es pugui reduir per tal que el nombre d'operacions realitzades sigui més petit.

Comentarem alguns resultats més generals segons[2]:

Per problemes de classificació, una combinació de funcions sigmoides dóna bons resultats.

La funcio ReLU és de les més usades perquè en molts casos funciona millor que d'altres funcions d'activació. Això fa que es comenci normalment per aquesta funció per investigar el desenvolupament d'un model, i si no dóna resultats satisfactoris es vagi canviant per les seves variacions (Leaky ReLU, etc...) o d'altres funcions d'activació.

A [1] ens diuen, però, que la tria d'una funció d'activació és important però que un cop hem 'entrenat' la xarxa neuronal la funció d'activació es pot canviar sense variar significativament els resultats de la xarxa neuronal.

5 Conclusions

Per finalitzar aquesta memòria, podem destacar vàries conclusions.

A mode de resum sobre el fet que les xarxes neuronals de propagació cap endavant siguin aproximadors universals de funcions direm que:

Per una banda, hem vist que la via Kolmogorov-Sprecher ens diu que les xarxes neuronals de propagació cap endavant amb tres capes, n components a la primera capa, $2n+1$ nodes a la segona capa i m nodes a l'última capa són aproximadors universals de funcions contínues de $\mathbb{R}^n$ a $\mathbb{R}^m$ sempre que la funció d'activació sigui monòtona creixent i de classe $Lip[\frac{ln2}{ln(2N+2}]$

Per altra banda, la segona via ens diu que les xarxes neuronals de propagació cap endavant són aproximadors universals de qualsevol funció mesurable sempre que la funció d'activació de la xarxa neuronal sigui una funció squashing.

Pel camí de la segona via passem per una conclusion que també és important. Ho fem mitcançant del Teorema de Stone-Weierstass. Les xarxes neuronals de propagació cap endavant amb una capa d'entrada, una de sortida i una oculta amb una funció contínua no constant com a funció d'activació aproximen qualsevol funció contínua de $\mathbb{R}^r$ a $\mathbb{R}$ sobre un compacte.

Cap d'aquestes demostracions ens dóna un mètode per calcular els diferents elements que constitueixen la xarxa neuronal.

Quedarien, doncs, per continuar investigant varies qüestions:

Em pregunto que passaria amb les funcions d'activació que no siguin squashing. S'hauria d'investigar si existeix alguna manera de demostrar si podrien aproximar qualsevol funció mesurable també.

Hem vist en concret una funció d'activació que no és monòtona creixent. Per tant, cap de les dues vies ens assegura que aproximi res; en canvi s'està fent servir per modelitzar certs problemes. Aquest fet també ha de tenir una explicació, ja sigui perquè funciona per un problema concret o si es pot generalitzar.

També em pregunto si és suficient amb aproximar funcions mesurables, és a dir si tots els problemes on s'apliquen les xarxes neuronals es poden reduir a modelitzar funcions mesurables.

Totes les demostracions sobre aproximació no ens diuen com determinar constructivament els pesos i biaixos de la xarxa neuronal. Hem vist al llarg de la memòria una forma de determinar-los mitjançant el Backpropagation, fent servir el descent del gradient estocàstic, però n'hi ha més mètodes per fer-ho. Es podria estudiar en que consisteixen aquests mètodes, quines aventatges aporten i quins inconvenients. Si són útils sempre o només seria interessant aplicar-los sota certes condicions, ja que degut a la seva complexitat ens poden donar molt més temps de càlculs i per tant ser menys eficaços.

S'hauria d'estudiar també la taxa d'aprenentatge per veure si hi ha alguna manera de triar-la per a que sigui més eficaç, com fer que arribi al mínim amb menys passos.

La determinació d'un nombre mínim de capes que hagi de tenir la xarxa neuronal per garantir cert grau d'aproximació seria també interessant. Aquí també hauríem de tenir en compte els error d'arrodoniment.

L'estudi i comparativa de les diferents funcions d'activació ens donaria també un al-

tre camp per continuar investigant. Si hi ha alguna que sigui més eficaç que altre per determinats problemes i per què.

Sobre les funcions d'activació també seria interessant un estudi sobre el problema del vanishing gradient, quan la xarxa canvia els pesos i biaxos molt lentament, és a dir, 'aprèn' molt lentament. S'haurien d'investigar les diferents maneres d'evitar-ho.

Totes aquestes preguntes són les que s'han quedat sense resposta en aquesta memòria i podrien servir per realitzar noves immersions en aquest món.

Durant l'estudi i preparació d'aquesta memòria he fet servir coneixements adquirits durant la carrera en assignatures d'Anàlisi Matemàtic, Topologia, Probabilitats i Càlcul Numèric. Durant el desenvolupament del projecte , a part d'aprendre a fer servir el Latex, he adquirit coneixements en Teoria de la Mesura, entre d'altres.

La tasca de recerca bibliogràfica he estat important, consultant molts recursos, llibres i investigacions publicades relacionades amb el tema. No totes apareixen a la bibliografia, doncs algunes simplement han reorientat la recerca cap a d'altres.

Ha estat molt divertit i enriquidor.

Referències

[1] Allwyn Jones, S.; Sibi, P.; Siddarth, P.: Analysis Of Different Activation Functions Using Back Propagation Neural Networks, *Journal of Theoretical and Applied Information Technology*, Vol. 43, No. 3, 31 de gener de 2013.

[2] Athaiya, Anidhya; Sharma, Siddhart; Sharma, Simone: Activation Functions in Neural Networks. , *International Journal of Engineering Applied Sciences and Technology* Vol. 4, Issue 12, pàg. 310-316.
`https://www.ijeast.com/papers/310-316,Tesma412,IJEAST.pdf`, Consulta: gener de 2021.

[3] Billingsley, Patrick: Probability and Measure, *John Wiley & Sons, Inc.*, 1995.

[4] Burden, R. L.; Faires, J. D.: Análisis Numérico, *Thomson Editores* , capítol 10.4, pàg 628-634.

[5] Dugundji, James : Topology. *Allyn and Bacon, Inc., Boston*, 1970.

[6] Gallant, A. R., White, H.: There Exists a Neural Network That Does Not Make Avoidable Mistakes., *In I.E.E.E. Second International Conference on Neural Networks* pàg. 657-664, 1988.

[7] Goldberg, Yoav : Neural Network Methods For Natural Language Processing (Synthesis Lectures on Human Language Technologies) , *Morgan and Claytoon Publishers*, 2017.

[8] Guilhoto, L.F.: An Overview Of Artificial Neural Networks for Mathematicians, `http://math.uchicago.edu/ may/REU2018/REUPapers/Guilhoto`, Consulta: gener de 2021.

[9] Halmos, Paul R.: Measure Theory, *Springer- Verlag*, 1974.

[10] Haykin, Simon: Neural Networks, A Comprehensive Foundation, *Prentice Hall International, Inc.*, 1999.

[11] Hetch Nielsen, R: Kolmogorov's Mapping Neural Networks Existence Theorem. , *Proceedings of The First I.E.E.E. International Conference On Neural Network*, pàg. 11-14, 1987.

[12] Hetch Nielsen, R.: Theory Of The Back Propagation Neural Networks. , *Proceedings of the International Joint Conference on Neural Networks*, pàg. 593-608, 1989.

[13] Hornik, K.; Stinchcombe, M.; White, H.: Multilayer Feedforward Networks Are Universal Approximators *Neural Networks,* , Vol 2, Issue 5, pàg 359-366, 1989.

[14] Kolmogorov, A. N.: On the Representation of Continuous Functions of Several Variables as Superposition of Continuous Functions of One Variable and Addition , *Dokl. Akad. Nauk SSSR 114:5*, pàg 953-956, 1957.

[15] Nielsen, M.: Neural Networks and Deep Learning, *Determination Press, 2015.*.
`http://neuralnetworksanddeeplearning.com/index.html`, Consulta: gener de 2021.

[16] Rudin, W.: Análisi Funcional. *Editorial Reverté, S. A.*, 1979.

[17] Sprecher, David A.: On the Structure of Continuous Functions Of Several Variables, *Transactions of the American Mathematical Society*, Març de 1965.