

Guixhe++ Biblioteca Computacional para Manejo de Redes Neuronales Artificiales

Daniel A. Cervantes Cabrera¹, Miguel A. Moreles Vázquez²

1 de Marzo, 2023

¹INFOTEC-CDMX.

²CIMAT-GTO.

Contenido

- ➊ Introduction al AP
- ➋ Generalización
- ➌ Solución numérica de EDP con AP
- ➍ Ghixhe++
- ➎ Ghixhe++
- ➏ Aproximación de campos vectoriales

Un problema de clasificación

Deep Learning: An Introduction for Applied Mathematicians, Catherine F. Higham.

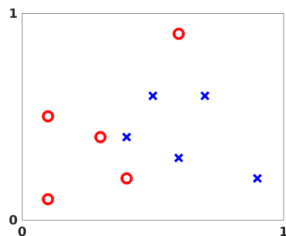


Figure: Círculos categoría A, Cruces categoría B.

Construir una función $F(\mathbf{x}) : (0, 1) \times (0, 1) \rightarrow \mathbb{R}^2$ tal que,

$$F(\mathbf{x}) = \begin{cases} A = (1, 0)^T \\ B = (0, 1)^T \end{cases}$$

Redes Neuronales Computacionales

Aprendizaje Profundo (*Deep Learning*): $F(x)$ es una “Red Neuronal Computacional”.

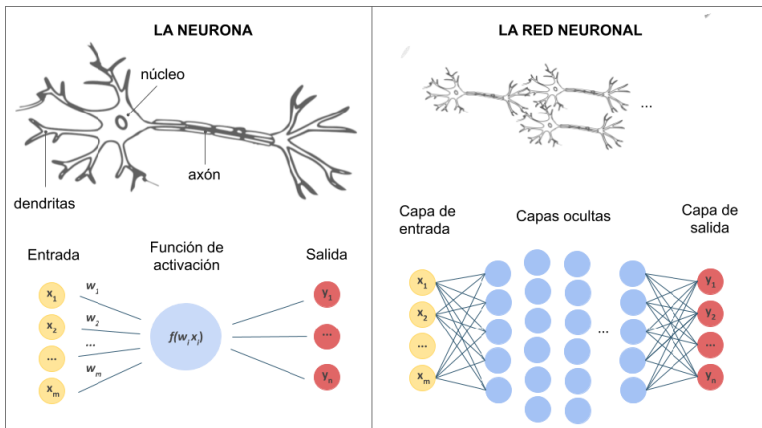


Figure: Modelo simplificado de Red Neuronal Computacional.

Función de Activación

Acción sobre el núcleo.

Para

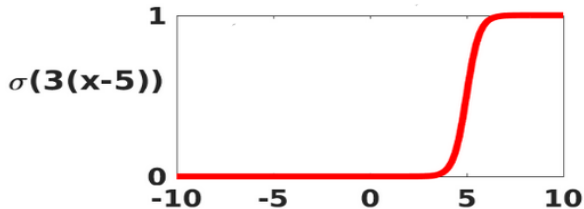
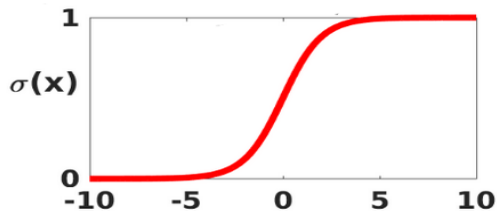
$$z \in \mathbb{R}^m \quad \sigma : \mathbb{R}^m \rightarrow \mathbb{R}^m,$$

$$(\sigma(z))_i = \sigma(z_i).$$

E.g. Sigmoide:

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

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

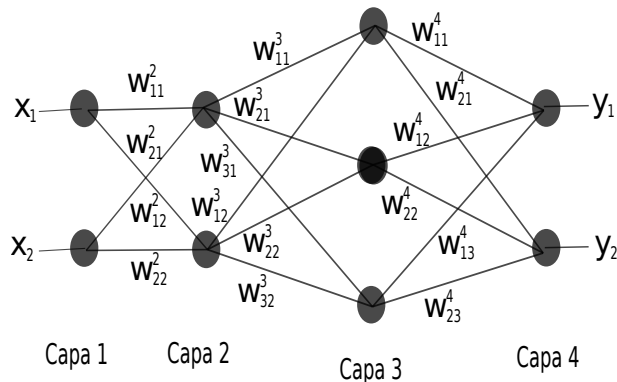


Ejemplo

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$W^{[2]} = \begin{pmatrix} w_{11}^2 & w_{21}^2 \\ w_{21}^2 & w_{22}^2 \end{pmatrix} \quad b^{[2]} = \begin{pmatrix} b_1^2 \\ b_2^2 \end{pmatrix}$$

$$\sigma(W^{[2]}x + b^{[2]}) \in \mathbb{R}^2.$$



$$F(x) = \sigma(W^{[4]}(\sigma(W^{[3]}\sigma(W^{[2]}x + b^{[2]}) + b^{[3]}) + b^{[4]}) \in \mathbb{R}^2.$$

Figure: Red Neuronal de cuatro capas.

Generalización

Dada una entrada $x \in \mathbb{R}^{n_1}$,

$$a^{[1]} = x \in \mathbb{R}^{n_1}$$

$$a^{[l]} = \sigma \left(W^{[l]} a^{[l-1]} + b^{[l]} \right) \in \mathbb{R}^{n_l} \text{ para } l = 2, \dots, L.$$

- Por determinar: $W = (W^{[1]}, W^{[2]}, W^{[3]}, W^{[4]})$, $b = (b^{[1]}, b^{[2]}, b^{[3]}, b^{[4]})$.
- Datos de entrenamiento: $\{x_i\}_{i=1}^N$ y $\{y(x_i)\}_{i=1}^N$ donde

$$y(x) = \begin{cases} (1, 0)^T & \text{si } x \text{ es } \textcolor{blue}{x} \\ (0, 1)^T & \text{si } x \text{ es } \textcolor{red}{o} \end{cases}$$

- Funcional de costo:

$$L(W, b) := \frac{1}{N} \sum_{i=1}^N \frac{1}{2} \|y(x_i) - a^{[L]}(x_i)\|_2^2$$

Objetivo: $L(W, b) \rightarrow 0$

Entrenamiento

- Supongamos que W y b se escriben en un vector p tal que $p \in \mathbb{R}^s$ así:

$$L : \mathbb{R}^s \rightarrow \mathbb{R}$$

- Método de optimización: “Gradiente Descendiente”.
- Si $p = (p_1, \dots, p_s)$ y $\Delta p = (\Delta p_1, \dots, \Delta p_s)$. Entonces

$$L(p + \Delta p) \approx L(p) + \sum_{r=1}^s \frac{\partial L(p)}{\partial p_r} \Delta p_r = L(p) + \nabla L(p)^T \cdot \Delta p$$

- Por lo tanto, $\Delta p = -\nabla L(p)$
- $p \rightarrow p - \eta \nabla L(p)$. En donde η , es la razón de aprendizaje.

Entrenamiento

- Definiendo,

$$L_{x_i}(W, b) = \frac{1}{2} \|F(x_i, W, b) - y(x_i)\|_2^2$$

entonces

$$\nabla L(W, b) = \frac{1}{N} \sum_{i=1}^N \nabla L_{x_i}(W, b).$$

- “Gradiente Descendiente Estocástico”: $i \in \text{random}\{1, \dots, N\}$.

$$p \rightarrow p - \eta \nabla L_{x_i}(W, b)$$

- Retropropagación. Para $2 \leq l \leq L$

$$\frac{\partial L}{\partial b_j^{[l]}} = \delta_j^{[l]}, \quad \frac{\partial L}{\partial w_{jk}^{[l]}} = \delta_j^{[l]} \sigma_k'(z^{[l-1]});$$

con

$$\delta^{[L]} = \sigma'(z^{[L]}) \cdot (\sigma(z^{[L]}) - y)$$

$$\delta^{[l]} = \sigma'(z^{[l]}) \cdot (W^{[l+1]})^T \delta^{[l+1]} \text{ para } 2 \leq l \leq L-1$$

$$\text{con } z^{[l]} = W^{[l]} \sigma(z^{[l-1]}) + b^{[l]} \in \mathbb{R}_l^n \text{ para } l = 2, \dots, L.$$

Predicción

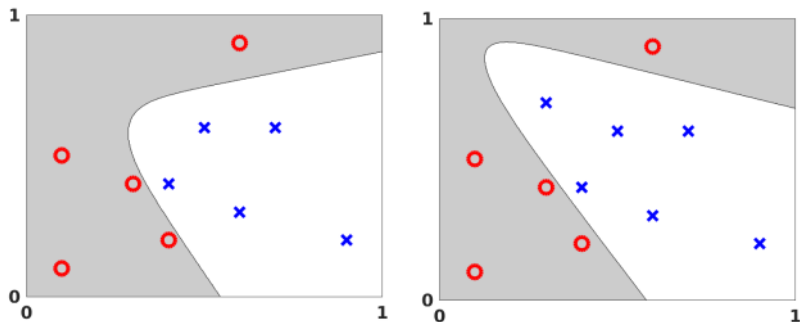


Figure: Clasificación dados los conjuntos de entrenamiento.

Solución numérica de EDP

Sirignano and Spiliopoulos (2018)

Sea u una función incógnita definida en $\Omega \times [0, T]$ con $\Omega \in \mathbb{R}^d$ la cual satisface la siguiente EDP

$$\begin{cases} (\partial_t + \mathcal{L})u(x, t) &= 0 & (x, t) \in \Omega \times [0, T] \\ u(x, 0) &= u_0(x) & x \in \Omega \\ u(x, t) &= g(x, t) & (x, t) \in \Omega \times [0, T] \end{cases}$$

Objetivo: Aproximar u con $F(x, t, W, b)$.

Solución numérica de EDP

- Una medida de qué tan buena es la aproximación del operador diferencial:

$$a := \|(\partial_t + \mathcal{L})F(x, t, W, b)\|_{\Omega \times [0, T]}^2$$

- Para la condición de frontera:

$$b := \|F(x, t, W, b) - g(x, t)\|_{\partial\Omega \times [0, T]}^2$$

- Para la condición inicial:

$$c := \|F(x, 0, W, b) - u_0(x)\|_{\Omega}$$

para una norma L^2 . De donde podemos definir el funcional de costo

$$L(W, b) = a + b + c$$

Algoritmo

- ① Inicializar los parámetros W, b y la razón de aprendizaje η .
- ②
 - Generar $ni = \{(x_i^n, t_i^n)\}_{i=0}^{N_{inner}}$ en $\Omega \times [0, T]$
 - Generar $nbdy = \{(z_i^n, \tau_i^n)\}_{i=0}^{N_{bdy}}$ en $\Omega \times [0, T]$
 - Generar $w = \{w_i^n\}_{i=0}^{N_{init}}$ en Ω
- ③ Calcular el funcional de costo $(x^n, t^n) = \{ni, nbdy, w\}$

$$L(x^n, t^n, W^n, b^n) = a + b + c$$

- ④ Actualizar los parámetros:

$$W^{n+1} = W^n - \alpha_n \nabla_{W^n} L(x_n, t_n, W_n, b_n)$$

$$b^{n+1} = b^n - \alpha_n \nabla_{b^n} L(x_n, t_n, W_n, b_n)$$

- ⑤ Repetir los pasos 2-4 hasta que los siguientes valores sean pequeños

$$\|W^{n+1} - W^n\| \|b^{n+1} - b^n\|$$

Teorema Sirignano and Spiliopoulos (2018)

- Si $L(W, b)$ el funcional de costo que mide la aproximación de una red neuronal de operador diferencial, de frontera y condiciones iniciales de una EDP.
- Sea \mathcal{C}^n la clase de redes neuronales con n capas ocultas.
- Si $f^n = \arg \min L(W, b)$.

Entonces

$$f^n \rightarrow u \text{ cuando } n \rightarrow \infty$$

Ghixhe++

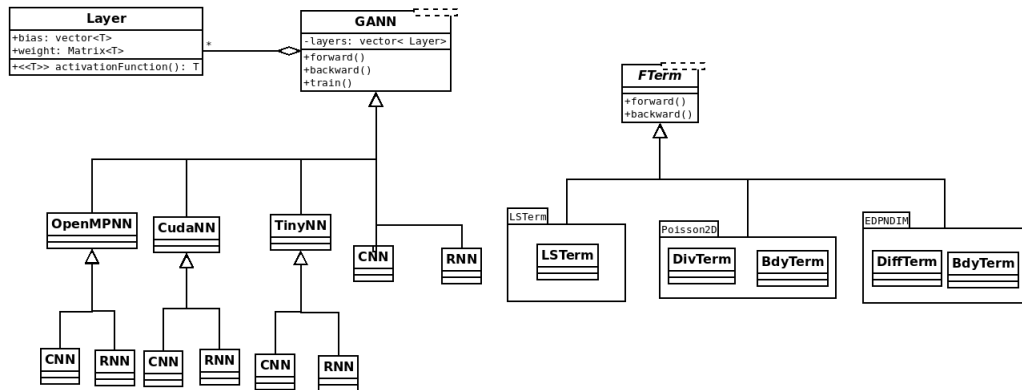


Figure: Diagrama de Clases UML

Ghixhe++




```
template <typename T>
class FTerm{
protected:
    NeuralNetwork<T>* _net;
public:
    virtual Matrix<T,1,Dynamic>* biasGradient(int)          = 0;
    virtual Matrix<T,Dynamic,Dynamic>* weightGradient(int) = 0;
    virtual void forward() = 0;
    virtual void backward() = 0;
    virtual void update() = 0;
    virtual void train() = 0;
};
```


Ghixhe++

```



FTerm<T> loss [N];
void train(FTerm<T> loss, int nMaxIt, T tol){
  for i=0,...,N-1
    loss[i].train();
  end
}
template <typename T>
void LSTerm<T>::train(){
  forward();
  backward();
  update();
}
template <typename T>
void LSTerm<T>::update(){
  int L = _architecture.size() - 2;
  for (int l = L; l >= 0 ; l--) {
    *FTerm<T>::_net->_b[l]-=FTerm<T>::_net->mLearningRate * *biasGradient(l);
    *FTerm<T>::_net->_W[l]-=FTerm<T>::_net->mLearningRate * *weightGradient(l);
  }
}



```







 **Ghixhe++** 
Project ID: 35838289 



87 Commits 2 Branches 0 Tags 14.6 MB Project Storage

Artificial Neuronal Network with C++, OpenMP and CUDA


 **test**
Daniel Cervantes authored 1 month ago 53a509a7 

main / ann / + Find file Web IDE  Clone 

 README  Add LICENSE  Add CHANGELOG  Add CONTRIBUTING  Add Kubernetes cluster  Set up CI/CD

 Add Wiki  Configure integrations

Name	Last commit	Last update
data	last commit	9 months ago
doc	test	2 months ago
examples	test	1 month ago
include	test	1 month ago
src	test	1 month ago
CMakeLists.txt	test	1 month ago
README.md	Update README.md	2 months ago

 README.md

Ghixhe++

Artificial Neuronal Network Library with C++, OpenMP and CUDA

Examples

Figure: Repositorio de Gitlab

Aplicación: Aproximación de campos vectoriales

- Dinámica de fluidos computacional.
- Aplicaciones en Meteorología.
 - Aproximación de velocidades de viento considerando condiciones físicas.
 - Dispersión de contaminantes.
 - Modelos de predicción de concentraciones de contaminantes de aire.
- Predicción de comportamiento en índices financieros
- Seguridad en IOT
- etc.

Problema

- Sea $\mathbf{u}^0 = \{(u_{1,i}(\mathbf{x}_i), u_{2,i}(\mathbf{x}_i), u_{3,i}(\mathbf{x}_i))\}_{i=1}^N$ un campo vectorial sobre una región acotada $\Omega \subset \mathbb{R}^3$ que representa un conjunto de observaciones de velocidades de viento en donde $u_{3,i}(\mathbf{x}) \equiv 0$.
- Se requiere construir una función $\mathbf{u}(\mathbf{x}) : \Omega \rightarrow \mathbb{R}^3$ que recupere la tercer componenete de \mathbf{u}^0 considerando condiciones físicas, como por ejemplo que sea incompresible ($\nabla \cdot \mathbf{u} = 0$) y/o irrotacional ($\nabla \times \mathbf{u} = 0$), etc.

Enfoque variacional

Supongamos que

-

$$\mathbf{E}(\Omega) := \{\mathbf{u} \in \mathbf{L}^2 := (L^2(\Omega))^n \mid \nabla \cdot \mathbf{u} \in L^2(\Omega)\}$$

- $J(\mathbf{u}) : \mathbf{E}(\Omega) \rightarrow \mathbb{R}^+$ tal que,

$$J(\mathbf{u}) = \frac{1}{2} \|\mathbf{u} - \mathbf{u}^0\|_{\mathbf{L}^2(\Omega), S}$$

-

$$\min_{\mathbf{u} \in \mathbf{E}(\Omega)} J(\mathbf{u}) \text{ sujeto a } \nabla \cdot \mathbf{u} = 0$$

Enfoque variacional

- $\mathbf{H} := \mathbf{E}(\Omega) \times L^2(\Omega)$ y $\mathcal{L} : \mathbf{H} \rightarrow \mathbb{R}$ dado por

$$\mathcal{L}(\mathbf{u}, \lambda) = J(\mathbf{u}) + \langle \lambda, \nabla \cdot \mathbf{u} \rangle_L^2(\Omega)$$

- Ecuaciones Euler-Lagrange

$$\begin{cases} -\nabla \cdot (S^{-1} \nabla \lambda) = \nabla \cdot u^0 & \lambda \in \Omega \\ \mathcal{B}\lambda = g & \lambda \in \partial\Omega \end{cases}$$

$$\hat{\mathbf{u}} = \mathbf{u}^0 + S^{-1} \nabla \hat{\lambda}(\mathbf{x})$$

Enfoque variacional

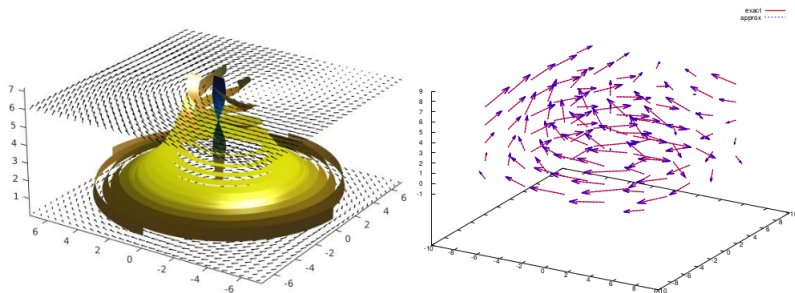


Figura 4.6: **Ejemplo 6.** Campo vectorial $\mathbf{u}(x, y, z) = (2ye^{\frac{-x^2-y^2}{49}} - \epsilon \frac{xz}{2}, -2xe^{\frac{-x^2-y^2}{49}} - \epsilon \frac{xz}{2}, \epsilon \frac{z^2}{2})$ con $\Omega = (-7, 7) \times (-7, 7) \times (0, 7)$ y $\epsilon = 0.1$ (derecha). Aproximación de \mathbf{u}_+ tomando $\mathbf{u}^0(x, y, z) = (2ye^{\frac{-(x^2+y^2+z^2)}{49}} - \epsilon \frac{xz}{2}, -2xe^{\frac{-(x^2+y^2+z^2)}{49}} - \epsilon \frac{yz}{2}, 0)$. Las flechas \rightarrow y \cdots , representan respectivamente el campo vectorial exacto y aproximado.

N	c	ϵ	$\kappa(G)$	$\nabla \cdot \mathbf{u}_+$	$\ \mathbf{u}_+ - \mathbf{u}\ _2 / \ \mathbf{u}\ $
27	0.01	0.1	1.508177e+13	2.633422e-03	1.463895e-01
125	0.01	0.1	2.632883e+21	4.063153e-03	4.732374e-04
512	0.01	0.1	8.866826e+21	1.365358e-02	5.872183e-05

Enfoque con AP

- Funcional de costo:

$$L(W, b) = \sum_{i=1}^N \frac{1}{2} \|(\mathbf{F}_1(\mathbf{x}_i, W, b) - \mathbf{u}_1^0)^2 + (\mathbf{F}_2(\mathbf{x}_i, W, b) - \mathbf{u}_2^0)^2\|_{L^2(\Omega)} + \\ \beta_1 \|\nabla \cdot \mathbf{F}(\mathbf{x}, W, b)\|_{L^2(\Omega)} d\mathbf{x} + \beta_2 \|F_3(\mathbf{x}, W, b)\|_{L^2(\partial\Omega)} dx_1 dx_2$$

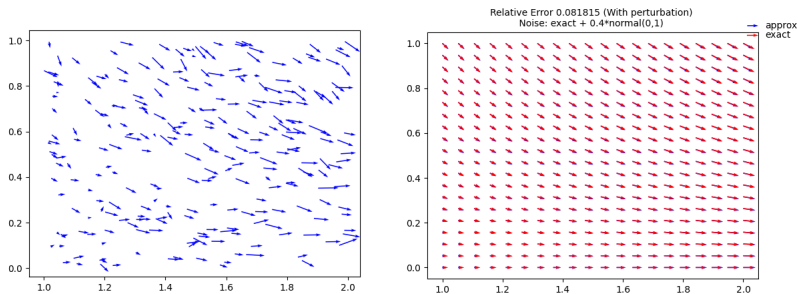


Figure: Aproximación 2D.

Enfoque AP

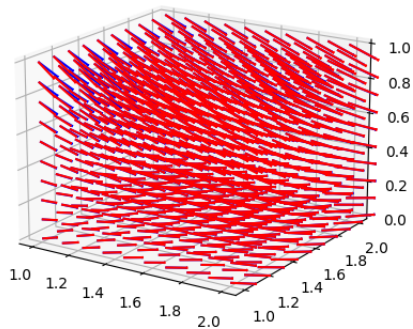
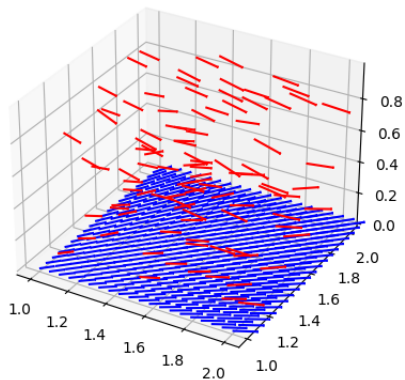
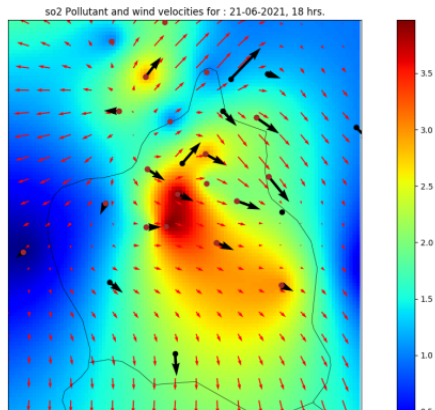


Figure: Aproximación 3D.

Enfoque AP: Trabajo Futuro

Finalmente se muestra la aproximación de los datos de contaminante y campos velocidades de viento de la Zona Metropolitana del Valle de México, con datos provenientes de estaciones meteorológicas del Zona Metropolitana del Valle de México. Una visualización de este aplicación actualizada en tiempo real se puede consultar en <https://rbfvectorapprox.matem.unam.mx:8050>.



Gracias

Gracias por su atención!!
daniel.cervantes@ciencias.unam.mx
daniel.cervantes@infotec.mx