U-tad
CENTRO UNIVERSITARIO
DE TECNOLOGÍA Y ARTE DIGITAL

Centro adscrito a
Universidad Camilo José Cela

Reservoir Computing

...

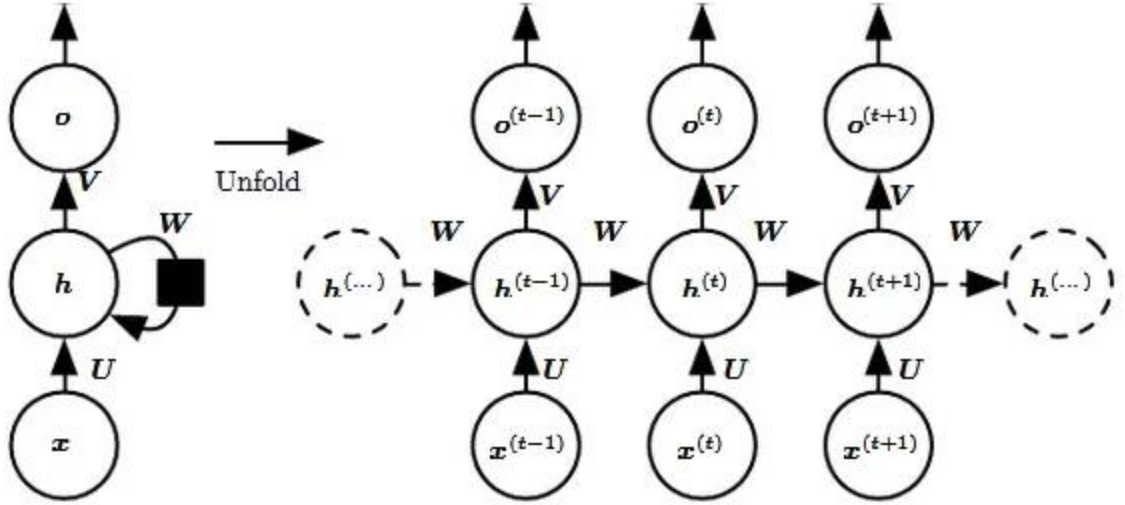
Origen de Reservoir Computing (RC).

El reservoir computing (RC) nace como una solución a la gran dificultad de entrenar redes neuronales recurrentes (RNNs) tradicionales.

Las RNNs porque poseen *memoria,* es decir, su estado actual depende de entradas y estados anteriores. Sin embargo, su entrenamiento es un problema de optimización no lineal complejo.

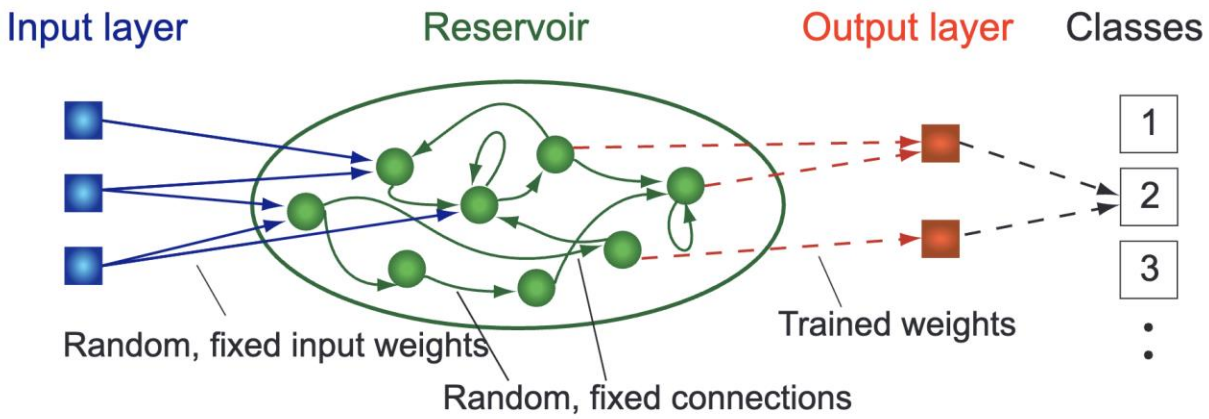
El método estándar sufre de **desvanecimiento y explosión de gradientes**. Esto significa que la señal de error se vuelve inmanejable (tendiendo a cero o a infinito) a medida que se propaga hacia atrás en el tiempo, haciendo casi imposible que la red aprenda dependencias a largo plazo.

El entrenamiento es, por tanto, computacionalmente carísimo y muy inestable.



En las RNNs tradicionales, entrenar todas las conexiones internas es una tarea muy compleja. RC evita esto, separando la dinámica del entrenamiento en dos componentes:

- El **reservorio**: Es una red neuronal recurrente grande, con conexiones internas fijas (generalmente aleatorias).
- La **capa de salida**: Es una capa lineal simple que lee el estado del reservorio.



Entonces, el proceso consiste en que **solo se entrenan las conexiones entre el reservorio y la capa de salida**. Dado que el reservorio es fijo y la capa de salida es lineal, el entrenamiento se convierte en un simple problema de regresión lineal, que es computacionalmente muy eficiente.

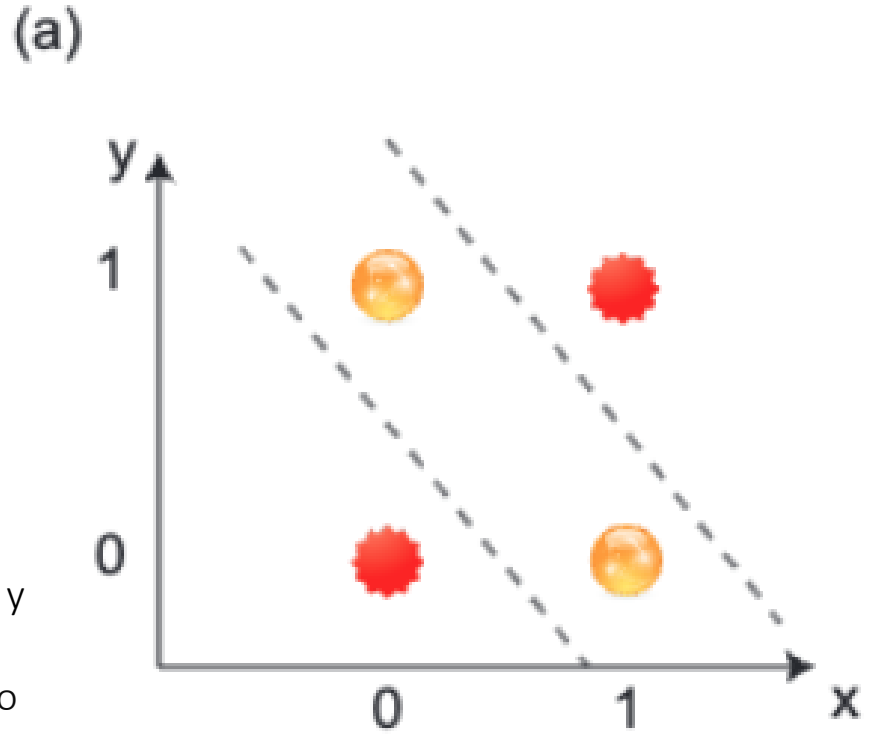
En la tesis se explica esta idea usando el **problema XOR** (Fig 1.5)

La función XOR tiene cuatro puntos de entrada en 2D (x1, x2):

- Clase 0 (estrellas): (0, 0) y (1, 1)
- Clase 1 (esferas): (0, 1) y (1, 0)

Como muestra la Fig 1.5 (a), este problema no es linealmente separable, es imposible trazar una única línea recta que separe las estrellas de las esferas.

Matemáticamente, es imposible encontrar un conjunto de pesos (w_1 , w_2) y un sesgo (b) que implementen correctamente la función XOR dado que no es linealmente separable, y un perceptrón simple [$sgn(w_1x_1 + w_2x_2 + b)$] solo puede resolver problemas linealmente separables.

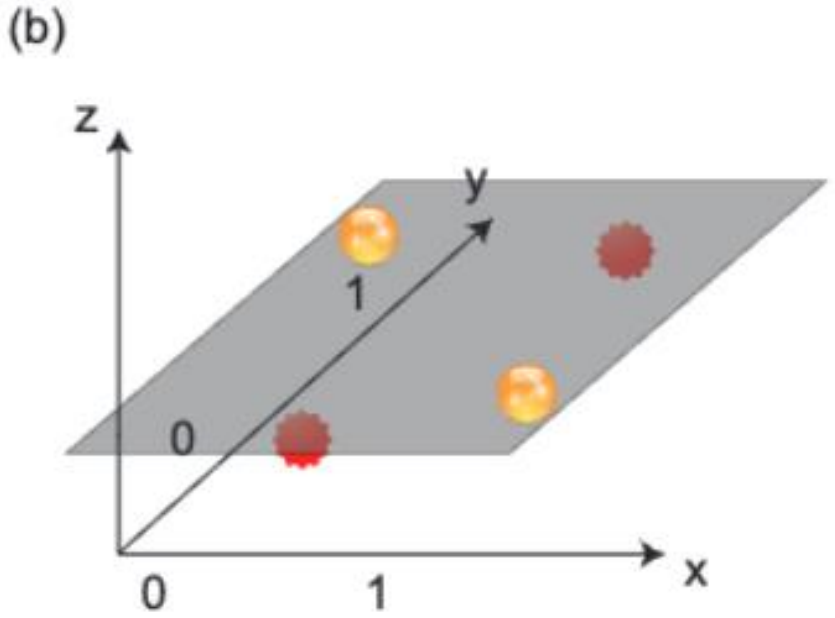


(Fig 1.5)

La solución sería aplicar una transformación no lineal para pasar 2D a 3D. Un ejemplo de mapeo podría ser:

$$\Phi(x_1, x_2) \rightarrow (z_1, z_2, z_3) = (x_1, x_2, x_1 \cdot x_2)$$

- $\Phi(0,0) \rightarrow (0,0,0)$ **[Clase 0]**
- $\Phi(1,1) \rightarrow (1,1,1)$ **[Clase 0]**
- $\Phi(0,1) \rightarrow (0,1,0)$ **[Clase 1]**
- $\Phi(1,0) \rightarrow (1,0,0)$ **[Clase 1]**



En este nuevo espacio 3D, los puntos son linealmente separables como se muestra en la Fig. 1.5 (b).

El reservorio realiza esta transformación no lineal automáticamente. Al tener muchos nodos *(N)* y dinámicas no lineales, mapea la entrada *u(k)* al vector de estado del reservorio *x(k)* de dimensión *N*

Entonces, para llegar a resolver el problema debemos tener en cuenta:

- Dinámica del reservorio, el estado del reservorio se actualiza mediante la siguiente ecuación:

$$x(k) = f(W_{res}^{res} \cdot x(k-1) + W_{in}^{res} \cdot u(k-1))$$

- Cálculo de la salida, la salida es simplemente una combinación lineal de los estados del reservorio.

$$\hat{y}(k) = W_{res}^{out} \cdot x(k)$$ (Simplificación de la ecuación 1.3 de la tesis).

Si recolectamos todos los estados x(k) en una gran matriz S (donde cada columna es un estado en un tiempo k), y todos los targets y(k) en una matriz Y, podemos resolver la ecuación:

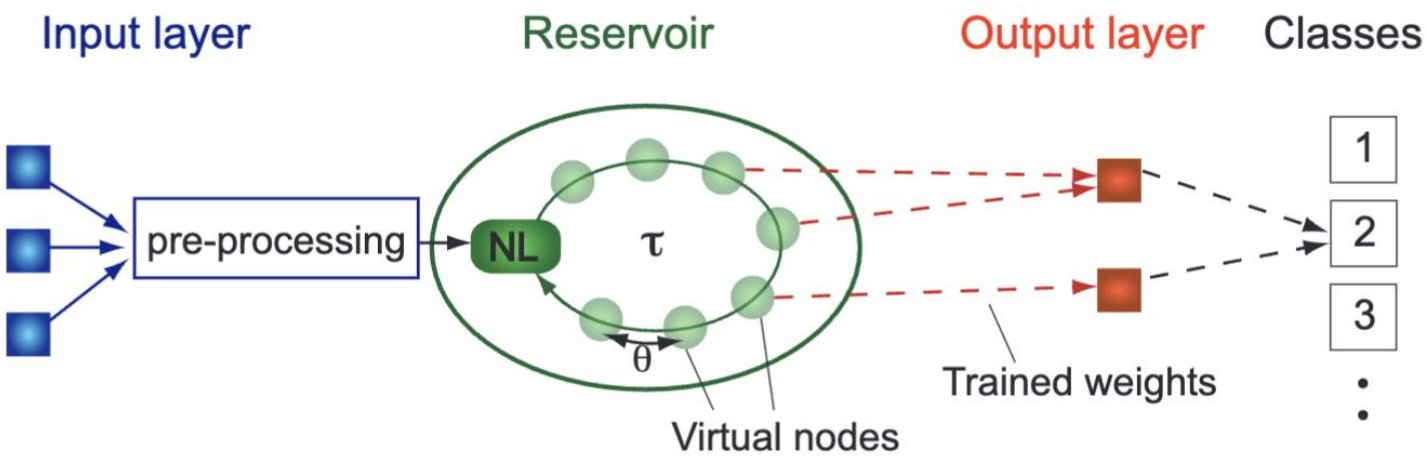
$$Y = W_{res}^{out} \cdot S$$

Entonces, la matriz de pesos de salida es un problema de regresión lineal.

Sistemas de retroalimentación retardada como reservorios.

Según la hipótesis central del punto 1.4 en la tesis, en lugar de usar una red de N nodos, se puede usar un solo nodo no lineal con un bucle de retroalimentación retardada para lograr el mismo resultado computacional.

La arquitectura propuesta consiste en un **solo nodo no lineal** (NL) y un bucle de **retroalimentación con un retardo τ.**



La idea clave es que un sistema dinámico descrito por una DDE posee un espacio de estados matemáticamente infinito.

¿Cómo un solo nodo puede emular una red?

En RC, cada nodo (i) recibe la entrada u(k) multiplicada por un peso diferente (w_{in} , I), pero en el caso de un solo nodo necesitaremos pre-procesar la entrada.

El pre-procesamiento se basa en tres conceptos:

- **Sample-and-Hold:** mantener la muestra de entrada u(k) constante durante todo un período de retardo (τ). I(t).
- **Máscara:** crear una función M(t), secuencia de N valores (pesos) diferentes. Esta máscara es periódica, con período τ.
- **Inyección:** Lo que realmente se inyecta en el nodo no lineal es el producto J(t) = I(t)·M(t) por cada canal de la entrada.

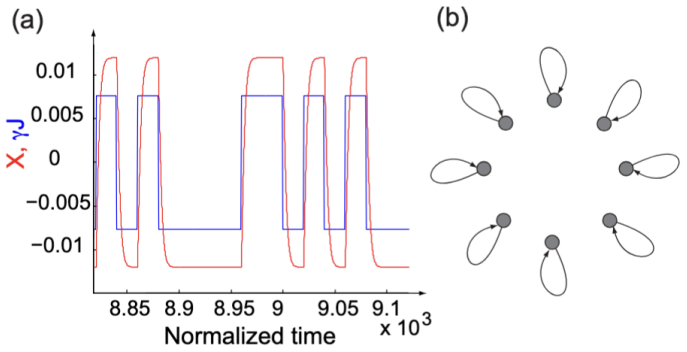
Durante el período τ en que la entrada u(k) es constante, cada "nodo virtual" (cada intervalo θ), ve esa misma entrada u(k) pero multiplicada por un valor *diferente* de la máscara M(t).

Al ser un único nodo la conexión no es física, sino **implícita**, creada por la inercia del nodo (el tiempo de respuesta).

El estado del nodo en un instante t (nodo virtual i), no solo depende de la entrada en t, sino también de su propio estado en el instante t – dt (nodo virtual i – 1).

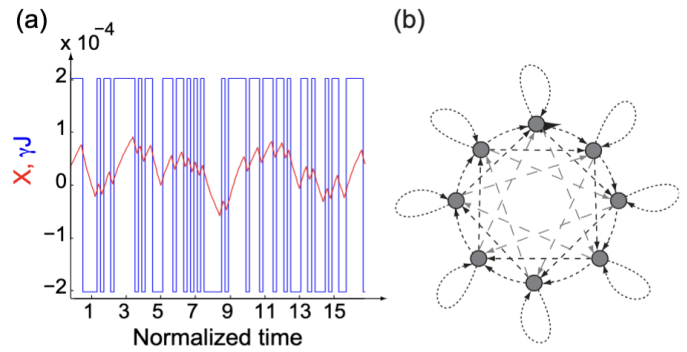
En la tesis se proponen dos estructuras de interconexión:

- **Sin conexión (Fig. 2.3):** Si el tiempo de respuesta del nodo (T) es mucho más rápido que la separación θ, el nodo se "resetea" por completo para cada nodo virtual. **Los nodos virtuales son independientes y no hay red.**



(Fig. 2.3)

- **Con conexión (Fig. 2.4).** Si el tiempo de respuesta del nodo es más lento (T > θ), el nodo *no* tiene tiempo de alcanzar el estado estacionario. Su estado para el nodo (i) depende del estado que tenía para el nodo (i – 1). **Esto crea una conexión recurrente entre nodos virtuales adyacentes**.



(Fig. 2.4)

Matemáticamente, en la tesis se modela el nodo no lineal con una ecuación diferencial general (normalizada a T = 1):

$$\dot{x}(t) = -x(t) + F[x(t-\tau), J(t)]$$ El término −x(t) representa la inercia del nodo.

Al resolver esta ecuación para el intervalo θ, obtenemos el estado del nodo virtual (i) en el paso de tiempo k ($x_{i,k}$):

$$x_{i,k} = x_{i-1,k} \cdot e^{-\theta} + (1 - e^{-\theta}) F(x_{i,k-1}, w_{in,i} u_k)$$

El término $e^{-\theta}$ es el peso de la conexión implícita entre el nodo virtual (i − 1) y el nodo (i).

Si aplicamos la recursión anterior repetidamente desde el inicio del ciclo hasta el nodo i , obtenemos una formulación que expresa el estado $x_{i,k}$ directamente en función de los estados del ciclo anterior (k - 1):

$$x_{i,k} = \Omega_i x_{n,k-1} + \sum_{j=1}^{i} \Delta_{ij} F(x_{j,k-1}, w_{in,j} u_k)$$

La Ω y la Δ son coeficientes que dependen de $e^{-\theta}$ y representan las conexiones ponderadas acumuladas desde el ciclo anterior hasta el nodo virtual (i) actual.

Equivalencia y entrenamiento

Ambas arquitecturas son computacionalmente equivalentes porque logran el mismo objetivo:

- Ambas mapean la señal de entrada que no es linealmente separable a un espacio de estados de alta dimensión donde el problema sí es linealmente separable.

- La red tradicional (RC) logra esto usando N (nodos físicos) en el espacio.

- El sistema retardado (DDS) logra esto usando N (nodos virtuales) en el tiempo, aprovechando el espacio de estados infinito de la DDE.

Conclusión, un solo nodo no lineal con retardo (NL), puede emular la dinámica de alta dimensión de una red completa.

En ambos casos, el entrenamiento al final es una regresión lineal:

- Se inyectan las entradas y se recolectan los estados de los N nodos (físicos o virtuales) en una matriz S.

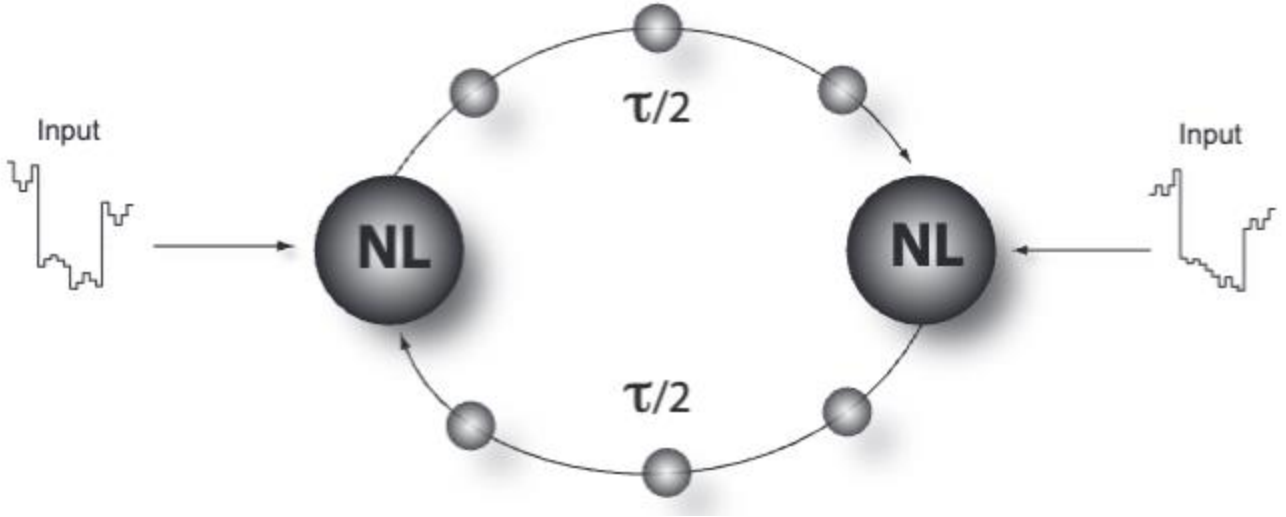
- Se resuelve la misma ecuación de regresión lineal para encontrar los pesos de salida.

Conclusión, las diferencias no están en el entrenamiento, sino en la arquitectura del reservorio.

Anexo:

El nodo único (NL) es simple pero lento, mientras que la red completa es paralela pero compleja.

En el punto 6.2 de la tesis se propone una arquitectura de 2 nodos (NL) con acoplamiento bidireccional.



Si se divide el trabajo, el tiempo de procesamiento por muestra se reduce a la mitad (pasa de τ a τ/2), lo que duplica la velocidad de entrenamiento.