



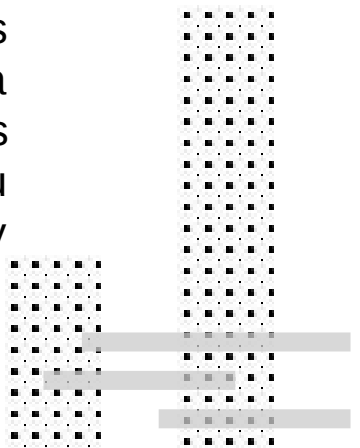
Time Series con DNN

Oscar Rodriguez

Advanced Time Series Analysis
27 Agosto 2025



Repaso



Revisamos las bases de las series de tiempo y los modelos clásicos de pronóstico (AR, MA, ARIMA, VAR, ETS), destacando la importancia de la **estacionariedad**, **tendencias**, **estacionalidades**, entre otros. Sin embargo, estos enfoques presentan limitaciones debido a sus supuestos de **linealidad** y necesidad de **ajustes manuales**, lo que motiva explorar métodos más flexibles. En esta sesión introduciremos las redes neuronales aplicadas a series de tiempo, desde arquitecturas simples como **MLP** hasta modelos especializados (RNN, LSTM, GRU, TCN o Transformers), comparando su desempeño con los métodos tradicionales en ejemplos univariados y multivariados.

MLP

Red neuronal básica –*Multi-Layer Perceptron*– de capas totalmente conectadas. Captura relaciones no lineales, pero no tiene memoria temporal..

RNN

Red neuronal –*Recurrent Neural Network*– que añade conexiones recurrentes (retroalimentación con estados pasados) que permiten mantener un "estado oculto" y procesar secuencias paso a paso, modelando dependencias temporales.

LSTM

Red neuronal –*Long Short-Term Memory*– Variante de RNN que introduce celdas de memoria y compuertas (input, output, forget) para manejar dependencias de largo plazo en secuencias.



GRU

–*Gated Recurrent Unit*– Variante más ligera de LSTM que combina compuertas de actualización y reinicio, con menos parámetros pero desempeño competitivo en muchas tareas secuenciales.

TCN


–*Temporal Convolutional Network*– Red convolucional 1D diseñada para secuencias. Usa convoluciones (filtro deslizante) para capturar dependencias de corto y largo plazo de manera paralela y más eficiente que las RNN.

Transformers

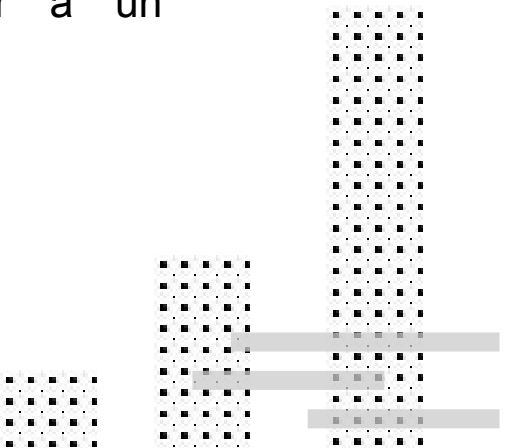
Tipo de red neuronal que usa mecanismos de atención en lugar de recurrencias o convoluciones, lo que les permite aprender relaciones entre todos los elementos de una secuencia de manera eficiente y precisa.



Requerimientos



Para generar los códigos de esta clase, es necesario instalar algunas librerías adicionales a las anteriormente usadas. Estas son `numpy=1.26.4`, `pmdarima`, `prophet`, `scikit-learn`, `TensorFlow=2.16.2` y acceder a un *Repositorio Series de Tiempo* creado para esta clase.



01 Activar el entorno

Activa el entorno con el comando
conda activate AdvTimeSeries

03 Actualiza

En la terminal ejecuta el comando:
*conda env update -n AdvTimeSeries
-f AdvTimeSeries.yml*

02 Descarga

Descarga el archivo
AdvTimeSeries.yml desde la URL:
<https://github.com/orodriguezm1/TimeSeries>

04 Códigos

Obtener los códigos disponibles desde:
<https://github.com/orodriguezm1/TimeSeries/tree/Dev>



MLP

Multilayer Perceptron (MLP)

Sea $x \in \mathbb{R}^d$ la entrada.

Un MLP con L capas ocultas se define como:

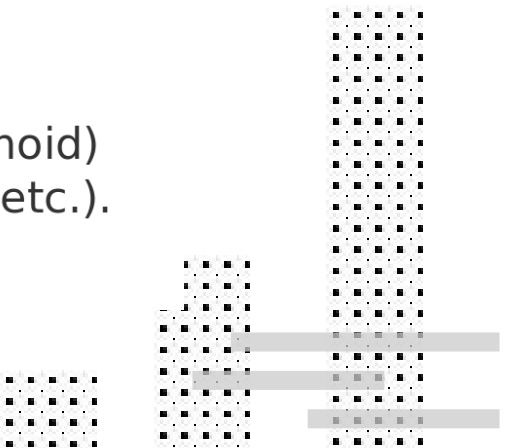
$$h^{(0)} = x$$

$$h^{(\ell)} = \sigma^{(\ell)}(W^{(\ell)}h^{(\ell-1)} + b^{(\ell)}), \quad \ell = 1, \dots, L$$

La salida es:

$$f(x) = \phi(W^{(L+1)}h^{(L)} + b^{(L+1)})$$

donde $\sigma^{(\ell)}$ son activaciones (ReLU, tanh, sigmoid)
y ϕ depende de la tarea (identidad, softmax, etc.).

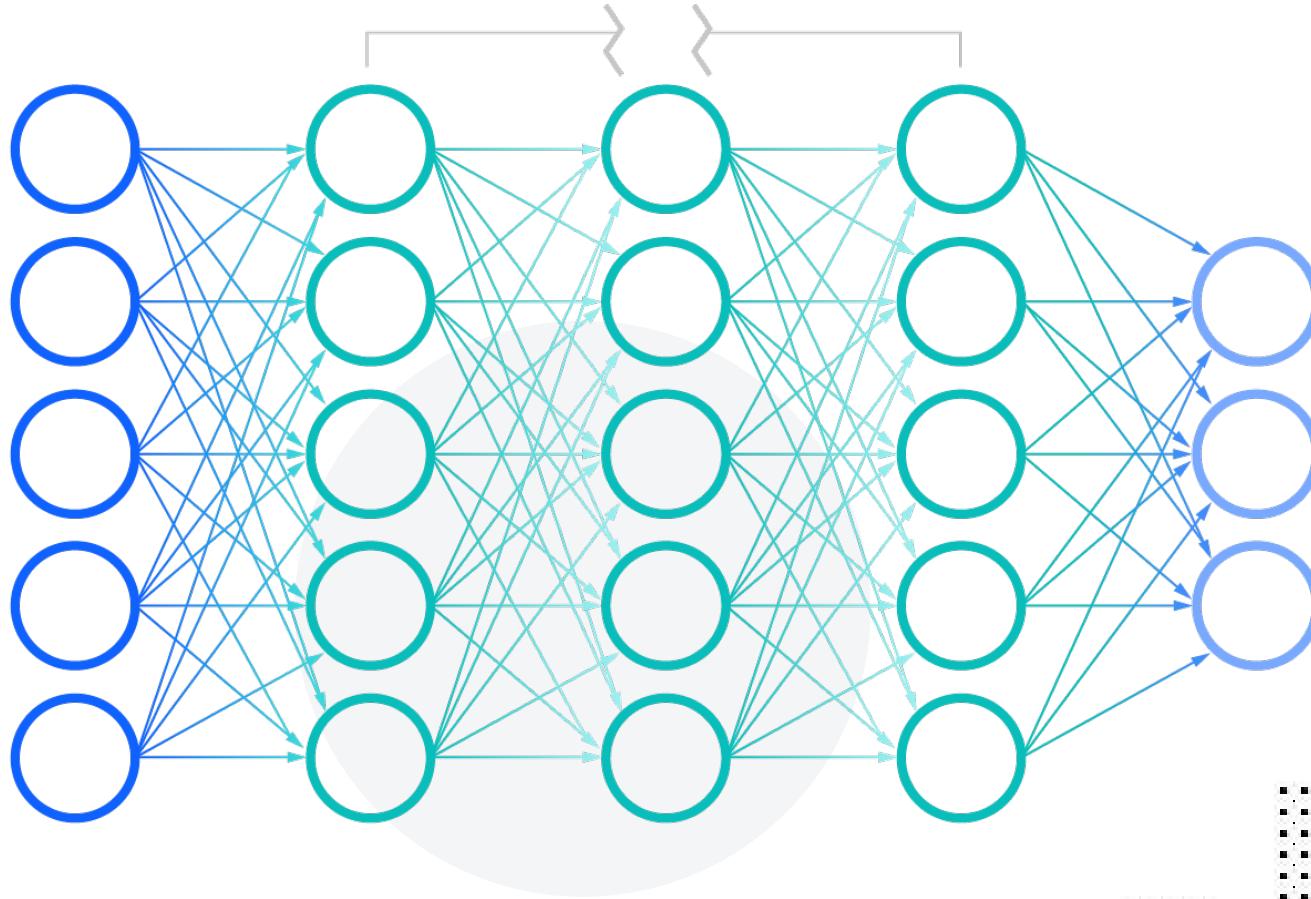


Deep neural network

Input layer

Multiple hidden layers

Output layer





RNN

Recurrent Neural Network (RNN)

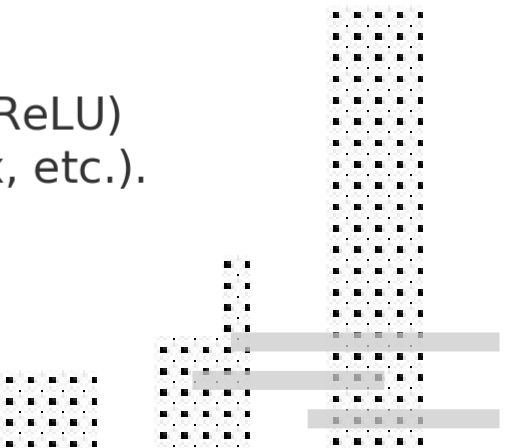
Sea $x_t \in \mathbb{R}^d$ la entrada en el tiempo t .
Un RNN mantiene un estado oculto $h_t \in \mathbb{R}^m$:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

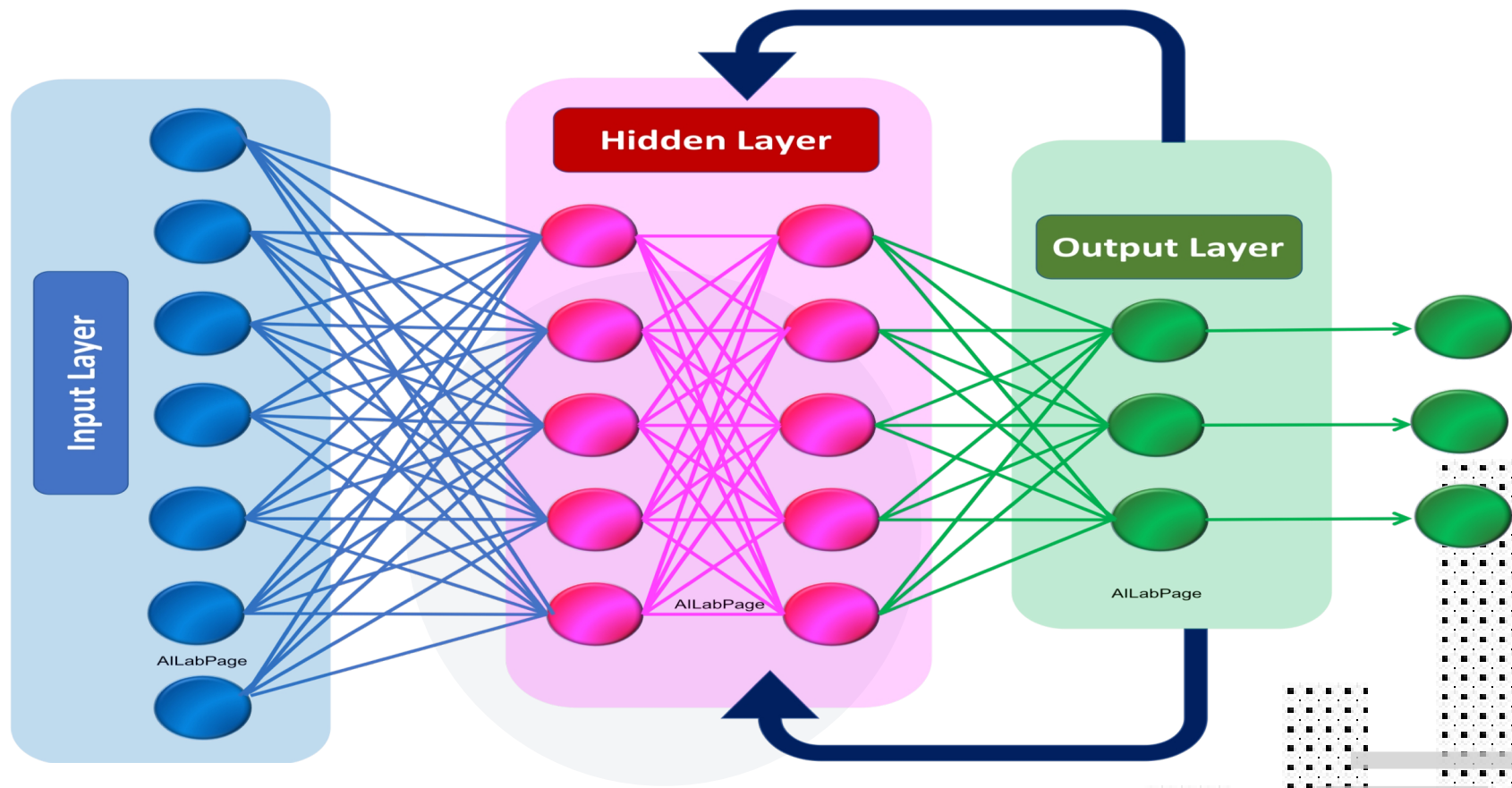
La salida en el tiempo t es:

$$y_t = \phi(W_{hy}h_t + b_y)$$

donde σ es la activación recurrente (tanh, ReLU)
y ϕ depende de la tarea (identidad, softmax, etc.).



Recurrent Neural Networks



LSTM

Long Short-Term Memory (LSTM)

Sea $x_t \in \mathbb{R}^d$ la entrada en el tiempo t ,
con estado oculto $h_t \in \mathbb{R}^m$ y estado de celda $c_t \in \mathbb{R}^m$.

Las ecuaciones son:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (\text{puerta de entrada})$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (\text{puerta de olvido})$$

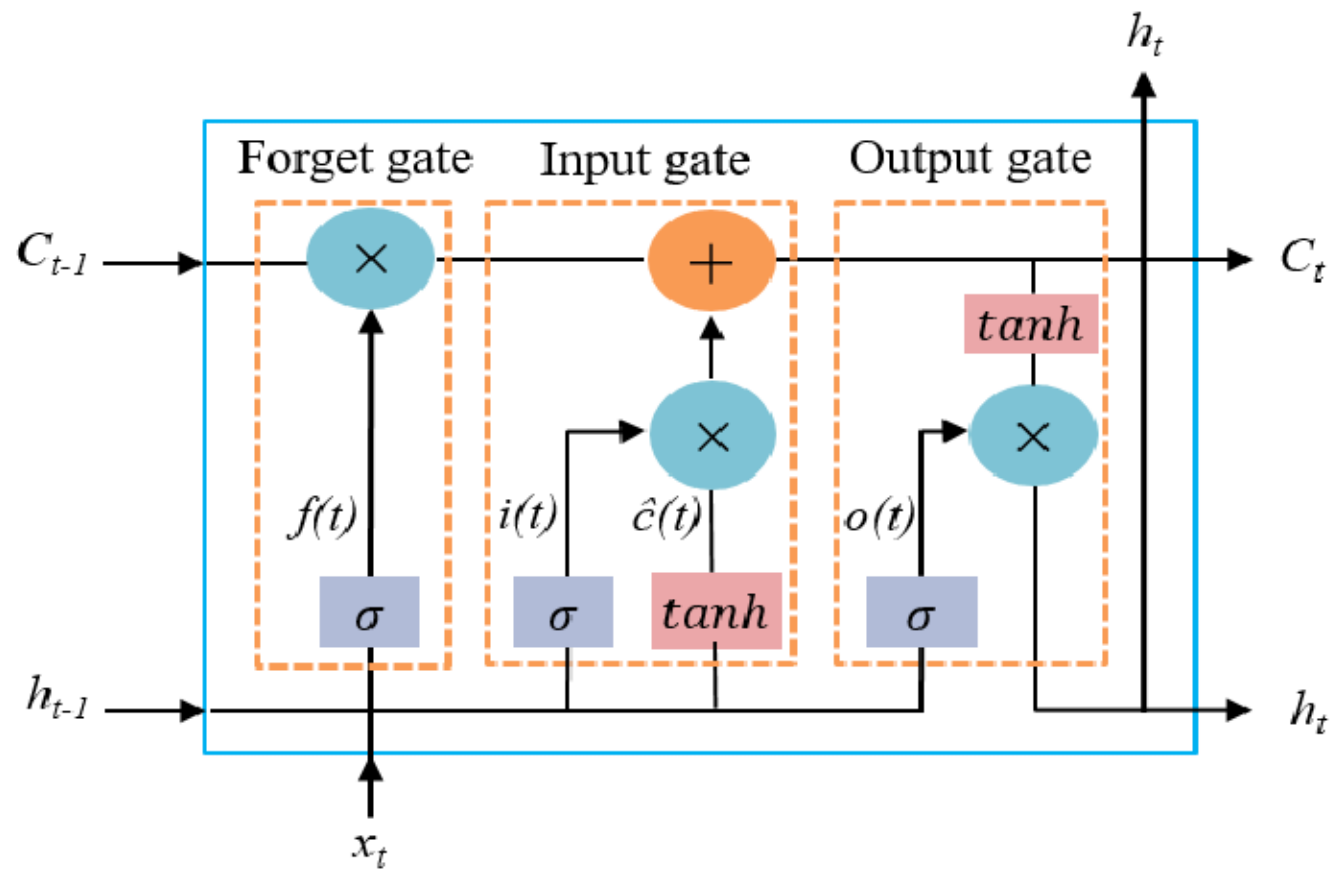
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (\text{puerta de salida})$$

$$\tilde{c}_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (\text{estado candidato})$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

$$h_t = o_t \odot \tanh(c_t)$$

donde σ es la sigmoide, \odot el producto elemento a elemento.

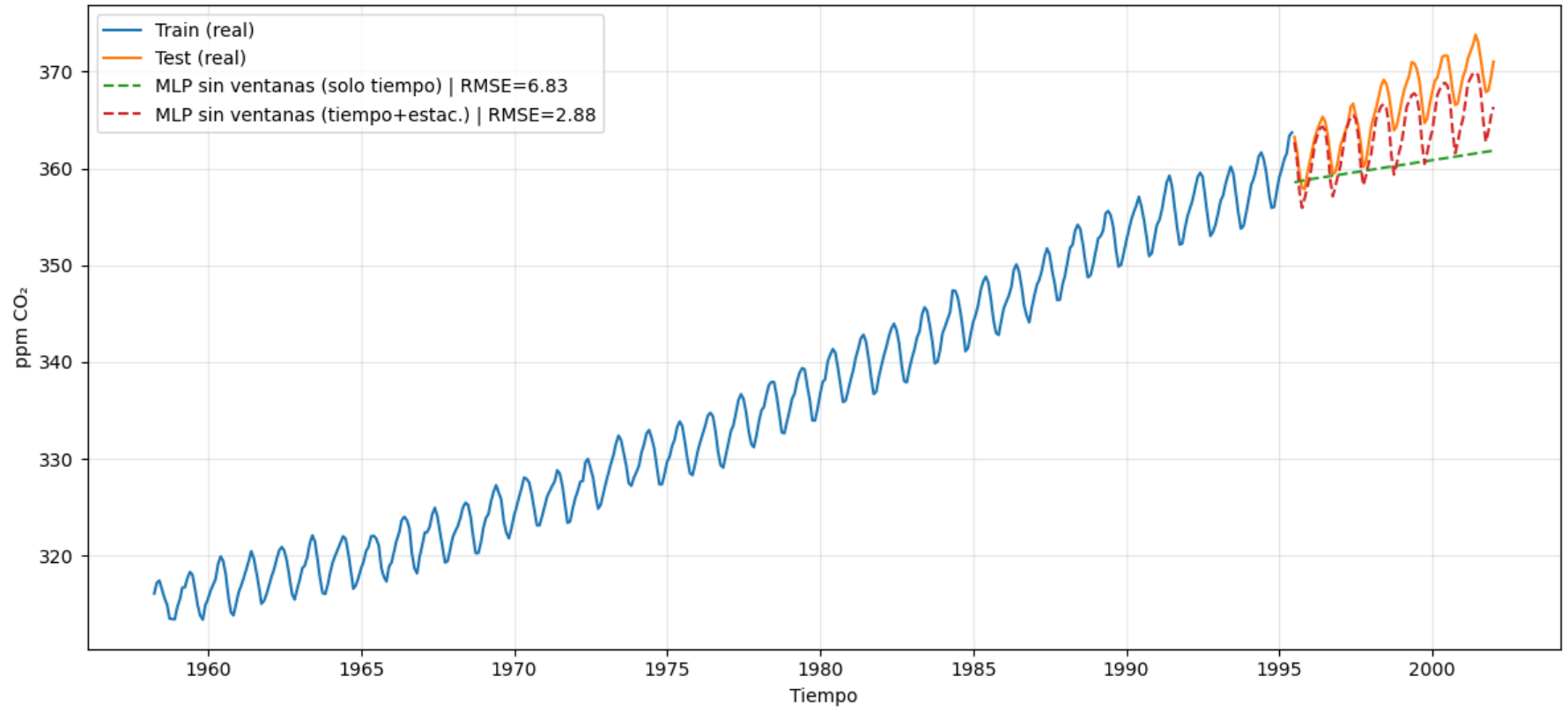




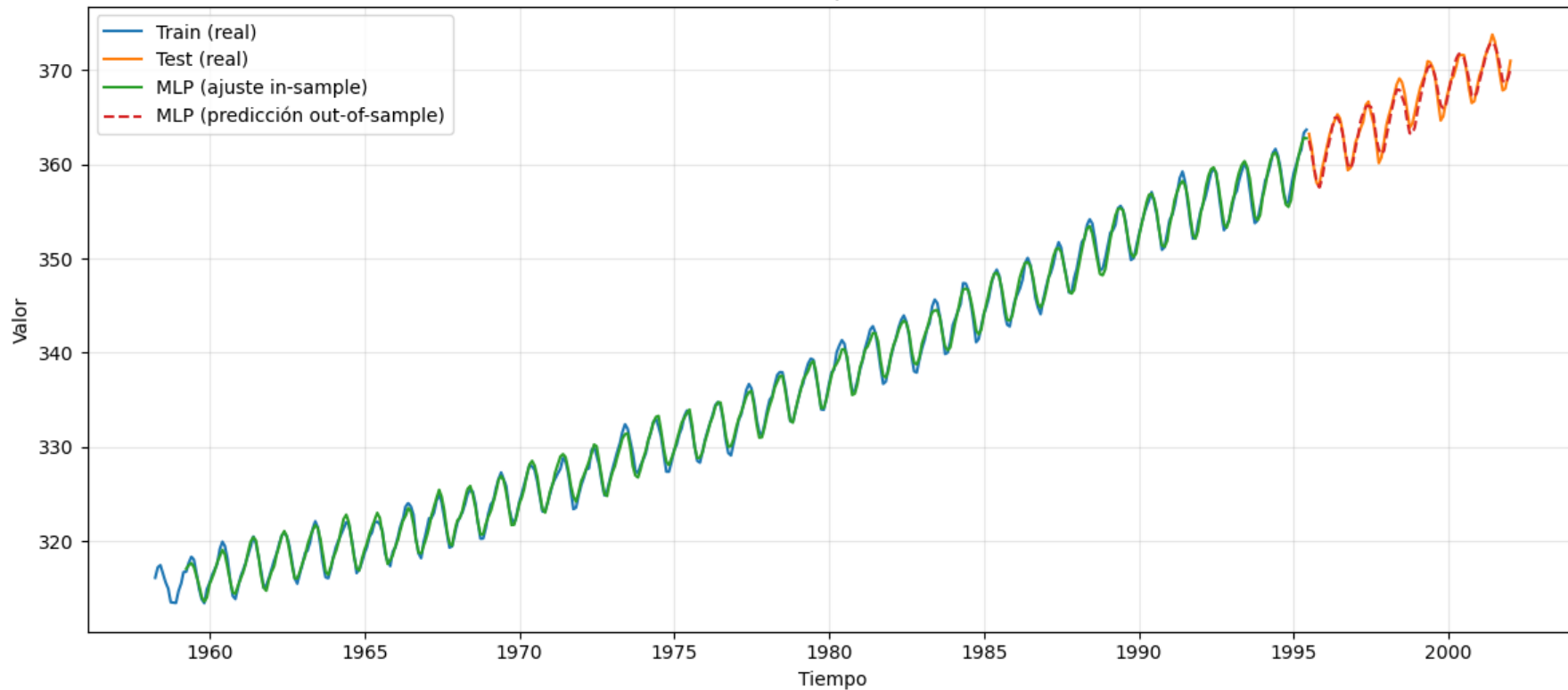
Ejemplos



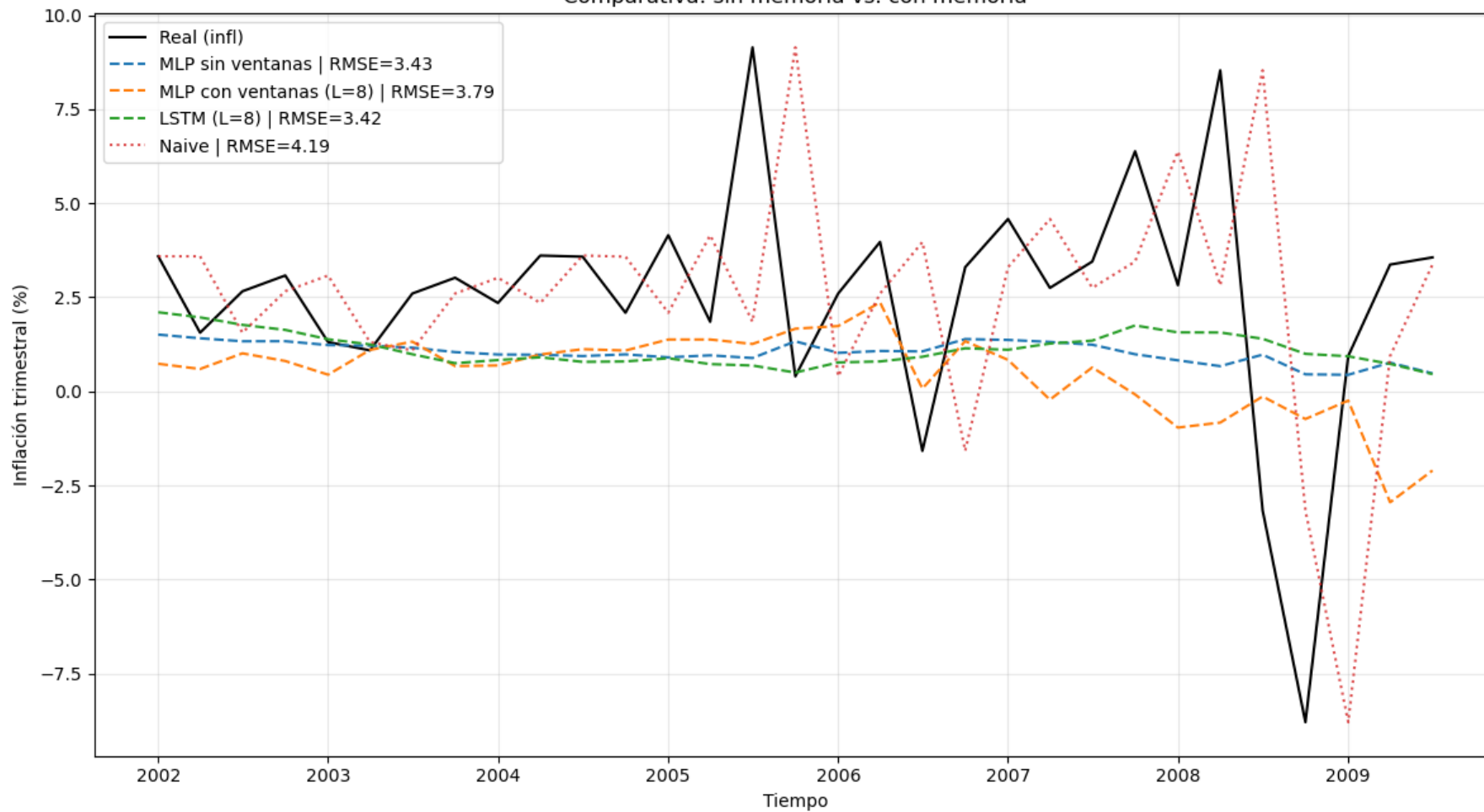
Pronóstico SIN ventanas deslizantes (MLP)
CO₂ mensual



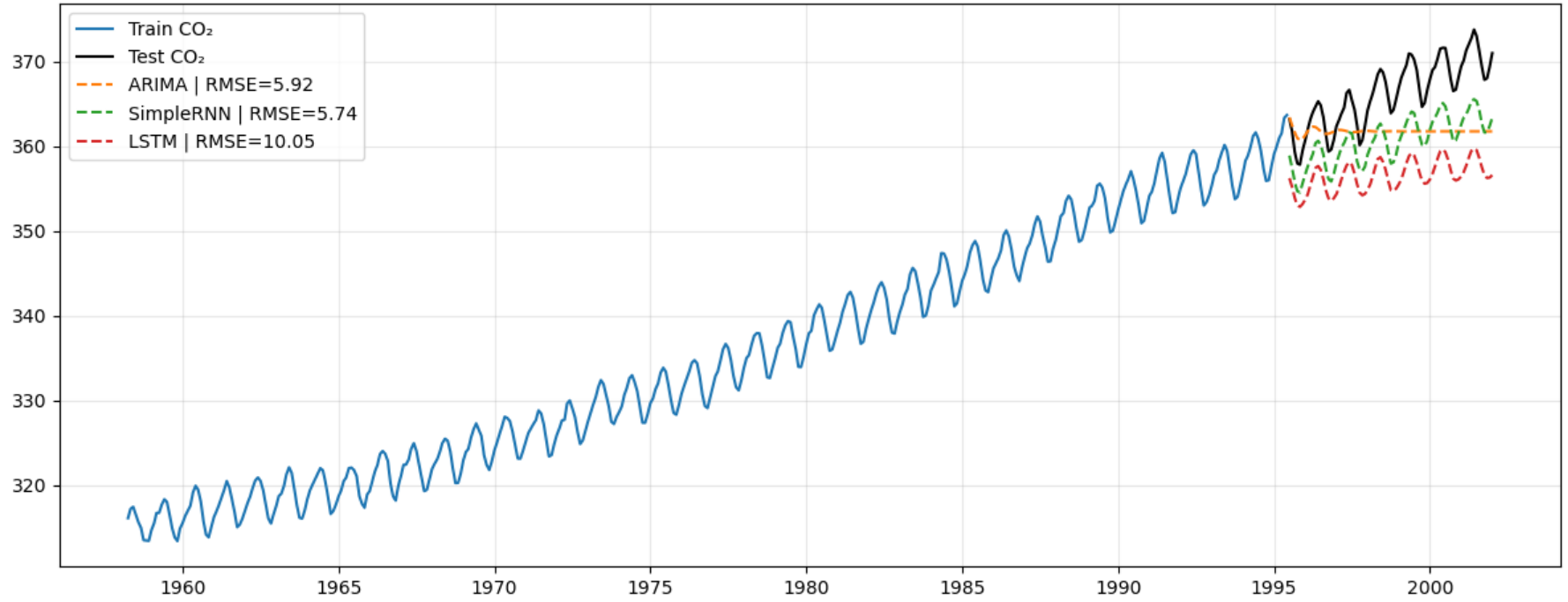
MLP para Serie Temporal (ventana p=12)
CO₂ (statsmodels, ppm mensual)
RMSE train=0.520 | RMSE test=0.707



Predicción de inflación trimestral (macro multivariado)
Comparativa: sin memoria vs. con memoria



CO₂ mensual — ARIMA vs SimpleRNN vs LSTM



Inflación trimestral — VAR vs LSTM multivariable

