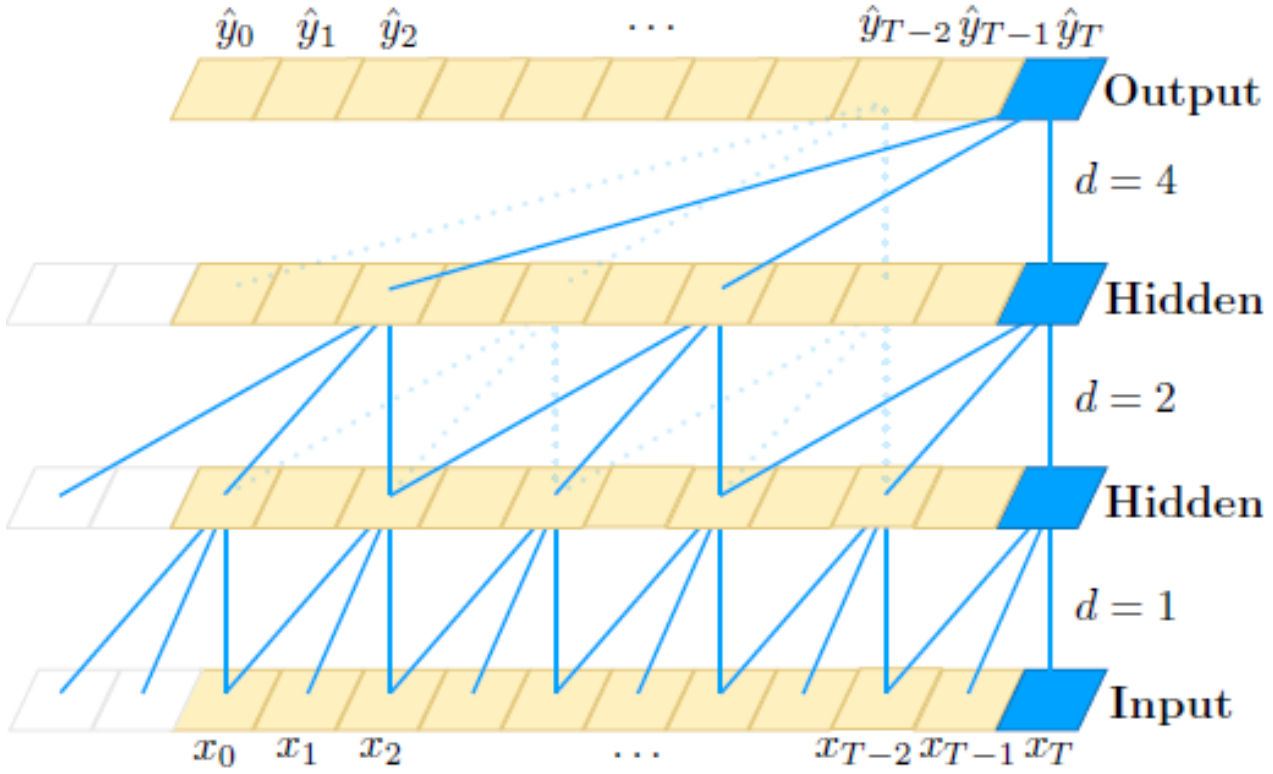
TEMPORAL CONVOLUTIONAL NETWORKS



Bai et al., 2018

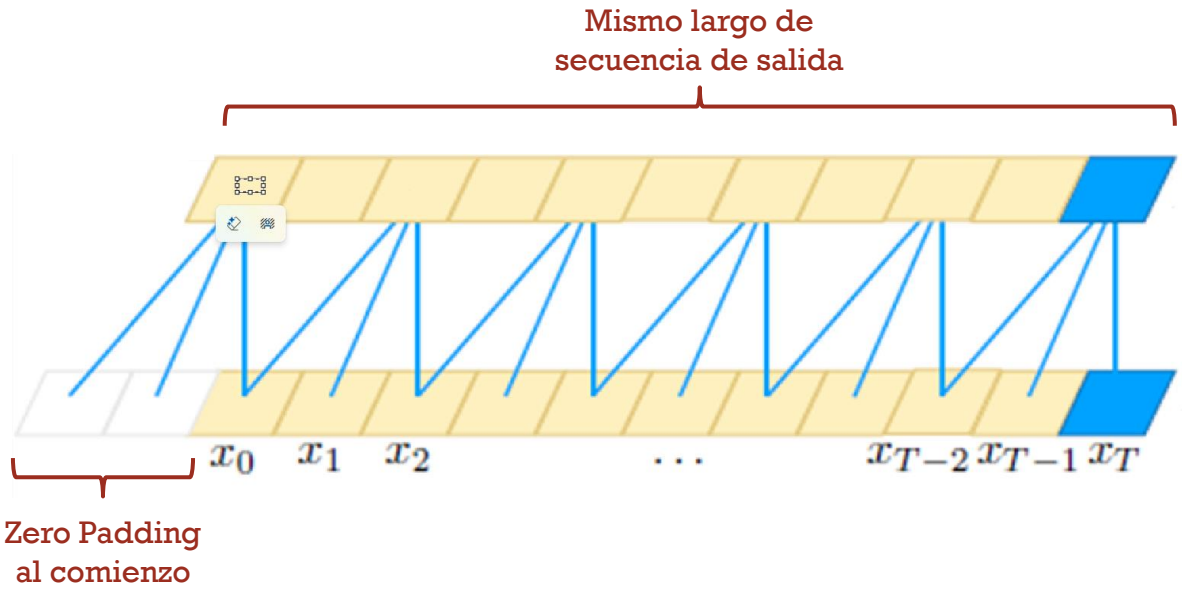
Profesor: Daniel Osorio

TEMPORAL CONVOLUTIONAL NETWORKS

- Las redes convolucionales temporales (**TCN – Temporal Convolutional Networks**) adaptan las redes convolucionales tradicionales (CNNs) a problemas de pronóstico a partir de datos secuenciales o series de tiempo
  - Data leakage al considerar datos futuros en el barrido con los kernels
  - Ampliación del campo receptivo
  - Superar el desvanecimiento del gradiente
- Superar problemas de redes recurrentes (LSTMs, GRUs):
  - Paralelización de los cómputos para reducir tiempos de entrenamiento
  - Reducción de requerimientos de memoria
  - Memoria de mayor alcance
  - Estabilidad de gradientes

CONVOLUCIONES CAUSALES

- Aprendizaje de patrones temporales sin acceder a información futura: las salidas dependen únicamente de las entradas presentes y pasadas (no hay data leakage) $f(\mathrm{t}+1) = f(t_0, \ldots, t_t)$
- Proceso de datos secuenciales 1D, uso de filtros convolucionales como en las CNNs tradicionales, mirando solo "hacia atrás"
- Largo de la secuencia de salida igual a la de entrada
- Zero Padding, sólo al comienzo de la secuencia



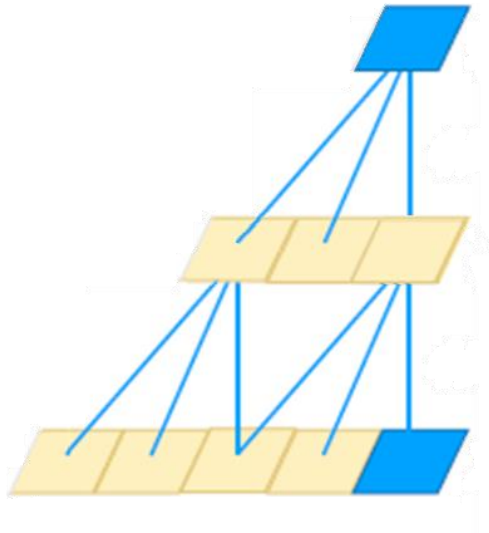
CAMPO RECEPTIVO

- Idealmente un pronóstico de un dato utiliza la información de todos los datos conocidos hasta ese punto temporal → el **campo receptivo** realiza un cubrimiento histórico completo
- Apilar n capas convolucionales permite ampliar el campo receptivo r al aplicar un kernel de tamaño k :

$$r = 1 + n * (k - 1)$$

- Dado un kernel k y una secuencia de tamaño l, se necesitan entonces n capas:

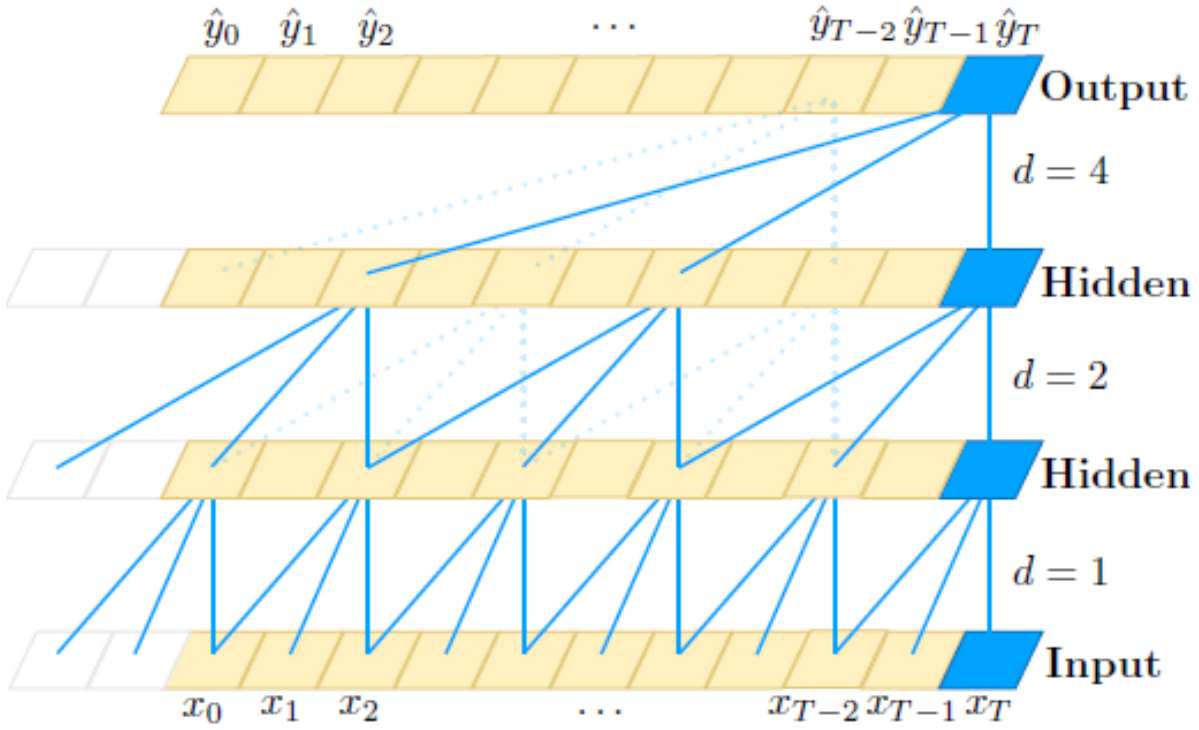
$$n = \lceil (l - 1)/(k - 1) \rceil$$



$$n = 2$$
$$k = 3$$
$$r = 1 + 2 * 2 = 5$$

DILATACIÓN

- Para evitar apilar demasiadas capas para poder lograr un cubrimiento histórico completo, se aplica el concepto de dilatación de las convoluciones en múltiples capas.
- Las secuencias de entrada a las capas más profundas tendrán saltos (*stride*) cada vez mas grandes entre ellos.
- Se establece un hiper parámetro b como base exponencial de la dilatación d a cada nivel de profundidad de las capas de convolución apiladas, tal que $d = b ** i$, siendo i el nivel de profundidad (empezando en 0).
- Con $b = 2$, los valores de d para varias capas apiladas serán entonces $1, 2, 4, 8, \ldots$ Una dilatación de 1 corresponde a secuencias de datos contiguos.



DILATACIÓN

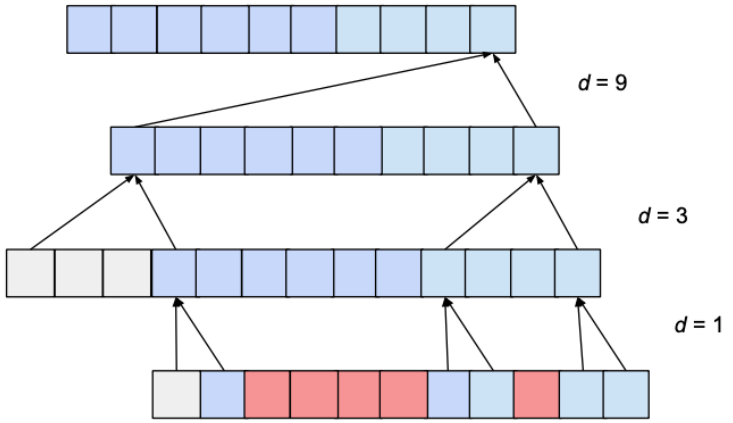
- Cada capa i necesitará un padding "causal" p para poder lograr el cubrimiento histórico completo

$$p = b^i * (k-1)$$

- Dado un kernel k, una base b, y n capas apiladas, el el campo receptivo máximo será:

$$r = 1 + (k-1) * \frac{b^n - 1}{b-1}$$

- Los valores de la base b y del largo de la secuencia l de entrada deben validar una restricción para que el cubrimiento sea completo y sin huecos.
  - Si $k \geq b$, no habrá problema
  - Si $k < b$, se debe cumplir que $1 + (k-1) * \frac{b^n - 1}{b-1} \geq l$
- Debido a la dilatación, el necesita suficiente información del pasado para capturar los patrones temporales y hacer predicciones precisas. La ventana de predicción futura debe ser más grande que la ventana histórica.

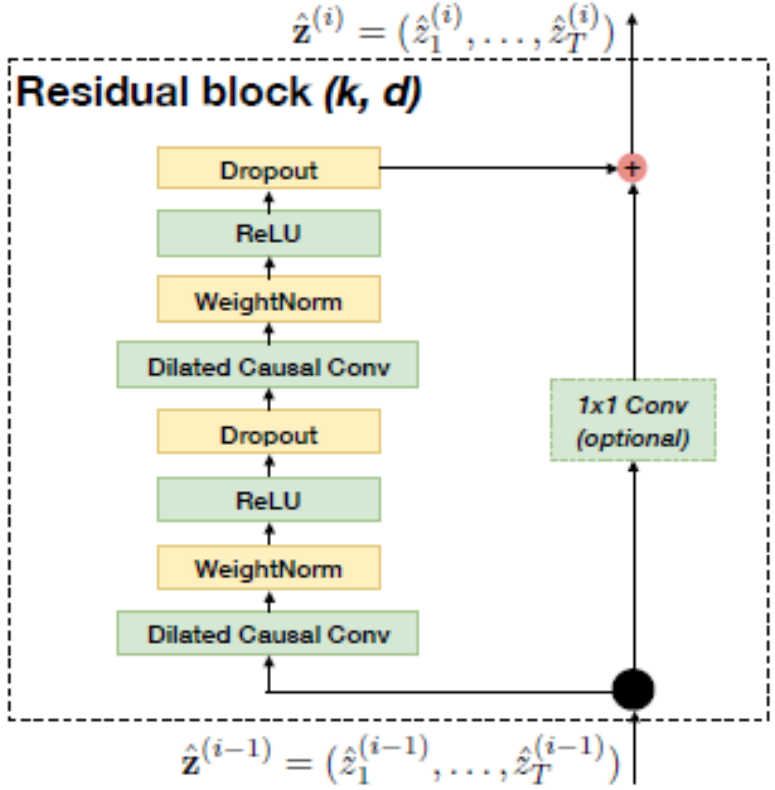


unit8.com

$$b = 3$$
$$k = 2$$

CONEXIÓN RESIDUAL

- Conservación de la señal de entrada, que se ira ajustando progresivamente bloque por bloque.
- Estabilidad de los gradientes durante el entrenamiento, permitiendo
  - Redes más profundas
  - Secuencias de mayores
- Uso opcional de una capa convolucional `1x1` para poder alinear la profundidad del tensor de entrada con la del tensor de salida.



Bai et al., 2018

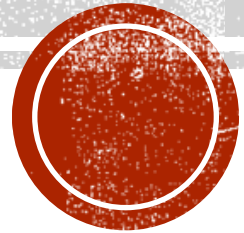
HIPER PARÁMETROS DE TCN

- **output_chunk_length**: Cantidad de pasos de tiempo que el modelo predice hacia futuro. Debe ser menor a la ventana histórica. Entre más grande, mas grande deberá ser la serie de tiempo de entrenamiento y más profunda la capa TCN.
- **input_chunk_length**: Longitud de la ventana histórica a considerar como base de una predicción. Entre mas grande el modelo podrá capturar mejor el contexto para realizar las predicciones. Usualmente un buen punto de partida es que sea entre 2 y 3 veces más grande que la ventana futura
- **kernel**: Tamaño del filtro convolucional. Usar filtros pequeños (2 o 3 pasos)
- **num_filters**: Número de filtros convolucionales en cada capa, cada uno de ellos capturando un patrón diferente e independiente.
- **dilation_base**: Controla el factor de dilatación de las convoluciones. Valores comunes: 2 o 3.
- **dropout**: hiperparámetro de regularización
- **num_layers**: Número de capas apiladas del bloque TCN.

COMPARACIÓN ENTRE TCN, LSTM Y GRU

Modelo	Ventajas	Desventajas
TCN	- Rápido y eficiente en paralelo. - Captura dependencias a largo plazo sin recurrencias. - Más robusto a la pérdida de memoria.	- Puede requerir más ajuste de hiperparámetros.
LSTM	- Maneja dependencias a largo plazo muy bien. - Amplia aplicación y soporte.	- Más lento de entrenar. - Computacionalmente intensivo.
GRU	- Más eficiente que LSTM (menos parámetros). - Buen balance entre rendimiento y costo.	- Menos flexible que LSTM en algunos casos.

GRACIAS



REFERENCIAS

- *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*, Shaojie Bai et al., 2018
- https://unit8.com/resources/temporal-convolutional-networks-and-forecasting/, Francesco Lassig, 2021
- *Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station,* in Methodologies and Application (24), Hewage et al., 2020