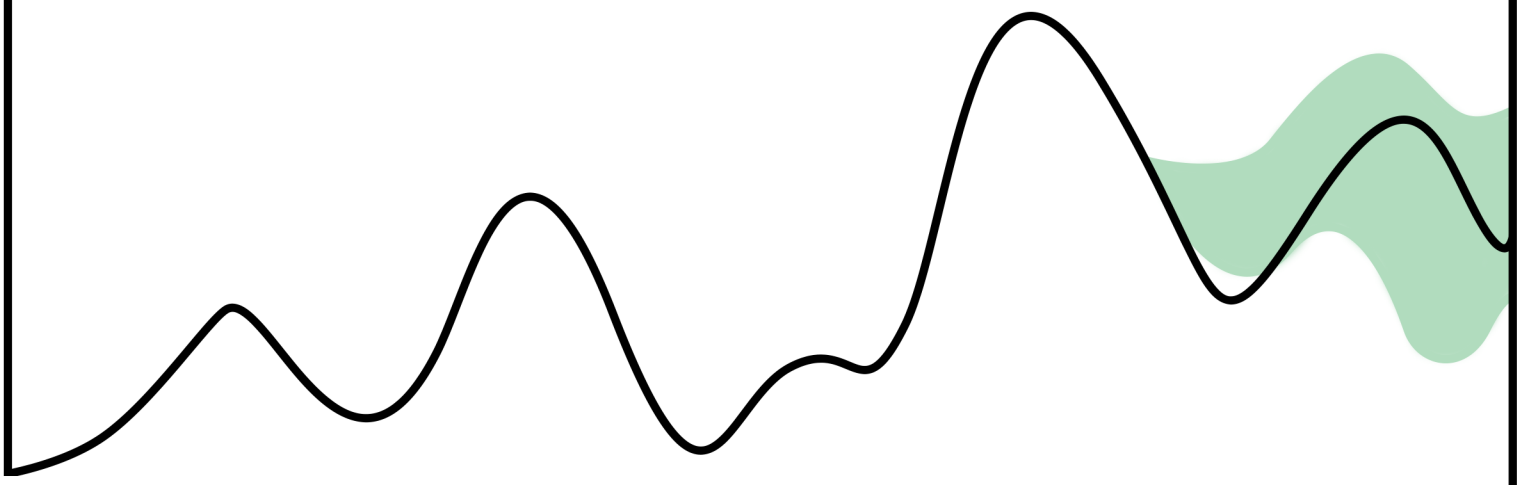
Comparación entre modelos tradicionales, de machine learning y deep learning para el pronóstico de series de tiempo

Anteproyecto de tesina



Alumno: Roncaglia Andrés

Directora: Mag. Méndez Fernanda

Carrera: Licenciatura en Estadística

UNIVERSIDAD NACIONAL DE ROSARIO
CONTINGERE HOMINEM COGITANTEM

UNR Universidad Nacional de Rosario

Tabla de contenidos

1. Introducción

Desde antaño el deseo de saber que traerá el mañana invade los pensamientos de las personas, quien no quisiera conocer los números ganadores del próximo 'Quini' o que acciones tendrán una importante subida para ganar mucho en muy poco tiempo. El análisis de datos ordenados en el tiempo puede ayudar a desenmascarar un poco de este futuro incierto, si bien no precisamente para ganar la quiniela o ganar mucho en el mercado accionista (aunque más de uno lo habrá intentado), sí puede colaborar en objetivos menos egoístas como encontrar los picos de consumo de luz para preparar los equipos y que no falle el sistema, o estudiar la ocupación de camas en hospitales para garantizar atención para todos, o tal vez analizar el clima para advertir a la población de temporales peligrosos. Estos ejemplos y muchos más prueban que tener una idea del futuro, o más bien una predicción, es crucial para la toma de buenas decisiones.

El estudio de las series de tiempo lleva años en desarrollo y permite realizar inferencias en datos temporales de diversas áreas, ya sea finanzas, medicina, medio ambiente, entre otras. Con un buen conocimiento de matemáticas, estadística e informática es sencillo hacer un pronóstico aproximado de casi cualquier dato que se mida en el tiempo.

Estos últimos años se vieron caracterizados por el gran aumento en los volúmenes de datos, el '*Big Data*' es presente y futuro, y en un mundo en donde todo se vuelve más complejo y el tiempo es cada vez más valioso, es conveniente tener herramientas que faciliten y acorten los tiempos de trabajo. Si bien los métodos actuales para trabajar series de tiempo son precisos, los modelos tradicionales como ARIMA requieren de un arduo trabajo manual que es dificil de automatizar y de amplios conocimientos para encontrar un buen ajuste, mientras que los métodos de *machine learning* que se utilizan actualemte pueden tomar un largo tiempo de entrenamiento y un gran coste computacional. Para resolver estos problemas llegan los modelos fundacionales pre-entrenados tales como *TimeGPT* o *Chronos*, que se encargan de buscar el mejor modelo para la serie especificada de manera automatizada, sin que el usuario tenga que hacer algún esfuerzo o tener algún conocimiento de como trabajar con datos temporales.

Si bien el objetivo de los modelos para series de tiempo está centrado en la predicción y no tanto así en la interpretación, es importante entender que está haciendo el modelo para justificar en cierta manera las predicciones que realiza. Es en este punto donde más pecan los modelos de aprendizaje profundo, entre los que se encuentran los modelos fundacionales, ya que no solo cuentan con un enorme número de parámetros, sino que también las metodologías usadas para el ajuste del modelo son muy complejas.

También es importante preguntarse hasta que punto mejoran las predicciones y los tiempos de ajuste por sobre los modelos más reconocidos actualmente en el análisis de series de tiempo, o si mejoran siquiera, ya que poco importa lo fácil que sea de realizar una predicción si la misma no es buena.

2. Objetivos

El objetivo de esta tesina es, en primer lugar, comparar la precisión, eficiencia y facilidad de pronosticar series de tiempo con modelos de *deep learning* en contraposición con otros métodos ya más establecidos como los tradicionales modelos ARIMA o modelos a partir del uso de *machine learning*.

Por otro lado también se busca que el lector obtenga conocimientos acerca de:

- Qué es una serie de tiempo y cúales son sus principales características
- Como funcionan los modelos que se comparan
- Que métricas se utilizan para comparar pronósticos
- Bajo qué condiciones un modelo funciona mejor que otro

Dado que naturalmente se necesitan datos que requieran ser pronosticados, otro objetivo propuesto es realizar predicciones sobre series univariadas de distintos ámbitos y características.

3. Metodología

Siendo el foco de la tesina la comparación de modelos complejos para series de tiempo, la mayor parte del documento estará centrada en la explicación del funcionamiento de estos y sus diversos parámetros ajustables. También será importante hacer mención a las características principales de una serie de tiempo y a las distintas formas que existen de comparar pronósticos.

3.1 Modelos tradicionales

Son llamados modelos tradicionales a aquellos que surgen antes del 'boom' del *machine learning* y los modelos de aprendizaje profundo. Son caracterizados por sus fuertes fundamentos estadísticos y su capacidad en capturar dependencias temporales en los datos.

3.1.1 ARIMA y SARIMA

Los modelos $ARIMA$ (*AutoRegresive Integrated Moving Average*) son unos de los modelos de pronostico tradicionales mejor establecidos. Son una generalización de los modelos autoregresivos (AR), que suponen que las observaciones futuras son combinaciones lineales de las p observaciones pasadas, y los modelos promedio móvil (MA), que pronostican las observaciones como funciones de los errores de las q observaciones pasadas. Además, generaliza en el sentido de los modelos diferenciados (I), en los que se resta a cada observacion los d-ésimo valores anteriores para estacionarizar en media, eliminando así las tendencias determinísticas.

Sin embargo este tipo de modelos no tienen en cuenta la posible estacionalidad que puede tener una serie, es por esto que se introducen los modelos $SARIMA(p,d,q)(P,D,Q)_s$ que agregan componentes AR, MA y diferenciaciones a la parte estacional de la serie con período s.

3.2 Modelos de *Machine Learning*

El *machine learning* se define como una rama de la inteligencia artificial enfocada a permitir que las computadoras y máquinas imiten la forma en que los humanos aprenden, para realizar tareas de forma autónoma y mejorar la eficiencia y eficacia a través de la experiencia y la exposición a mas información. Si bien los métodos que se presentan no fueron diseñados especificamente para el análisis de datos temporales, como los modelos tradicionales o aquellos que utilizan aprendizaje profundo que se mencionarán más adelante, si probaron ser útiles a lo largo del tiempo y a través de distintas pruebas.

Los métodos de machine learning a diferencia de los modelos tradicionales se enfocan principalmente en identificar los patrones que describen el comportamiento del proceso que sean relevantes para pronosticar la variable de interés, y no se componen de reglas ni supuestos que tengan que seguir. Para la identificación de patrones, estos modelos requieren la generación de características.

Es importante remarcar que lo que se presenta a continuación como modelos, no son más que técnicas de *boosting* aplicadas a modelos de bosques aleatorios. El concepto de *boosting* es crear modelos de forma secuencial con la idea de que los últimos modelos corrijan los errores de los previos.

3.2.1 XGBoost

XGBoost construye árboles de forma secuencial donde cada nuevo árbol busca predecir los residuos de los árboles anteriores. Es así entonces que el primer árbol buscará predecir los valores futuros de la serie, mientras que el segundo intentará predecir los valores reales menos los pronosticados por el primer árbol, el tercero tratará de inferir la diferencia entre los valores reales y el valor pronosticado del primer árbol menos los errores del segundo, y así sucesivamente.

Sin embargo, los modelos no se construyen infinitamente, sino que XGBoost busca minimizar una función de pérdida que incluye una penalización por la complejidad del modelo, limitando así la cantidad de árboles que se producen.

3.2.2 LightGBM

LightGBM funciona de una forma similar a XGBoost, la diferencias radican en la forma en que cada uno identifica las mejores divisiones dentro de los árboles y de que forma los hacen crecer. Mientras que XGBoost usa un método en el que se construyen histogramas para cada una de las características generadas para elegir la mejor división por característica, LightGBM usa un método más eficiente llamado *Gradient-Based One-Side Sample* (GOSS). GOSS calcula los gradientes para cada punto y lo usa para filtrar afuera aquellos puntos que tengan un bajo gradiente, ya que esto significaría que estos puntos están mejor pronosticados que el resto y no hace falta enfocarse tanto en ellos. Además, LightGBM utiliza un procedimiento que acelera el ajuste cuando se tienen muchas características correlacionadas de las cuáles elegir.

A la hora de hacer crecer los árboles, XGBoost los hace nivel a nivel, es decir que primero se crean todas las divisiones de un nivel, y luego se pasa al siguiente, priorizando que el árbol sea simétrico y tenga la misma profundidad en todas sus ramas. LigthGBM, en cambio, se expande a partir de la hoja que más reduce el error, mejorando la precisión y eficiencia en series largas, pero arriesgandose a posibles sobreajustes si no se limita correctamente la profundidad de los árboles.

3.3 Modelos de aprendizaje profundo

El *Deep learning* (aprendizaje profundo) es una rama del *machine learning* que tiene como base un conjunto de algoritmos que intentan modelar niveles altos de abstracción en los datos usando múltiples capas de procesamiento, con complejas estructuras o compuestas de varias transformaciones no lineales.

Entre el conjunto de algoritmos que se menciona están las redes neuronales, las cúales son un tipo de modelo que toma decisiones de la misma forma que las personas, usando procesos que simulan la forma biologica en la que trabajan la neuronas para indentificar fenómenos, evaluar opciones y llegar a conclusiones. Una red neuronal funciona con varias neuronas de entrada y salida, y distintos pesos. La suma de los pesos y las neuronas que no formen parte de la capa de entrada dan el total de parámetros que tiene que ajustar el modelo.

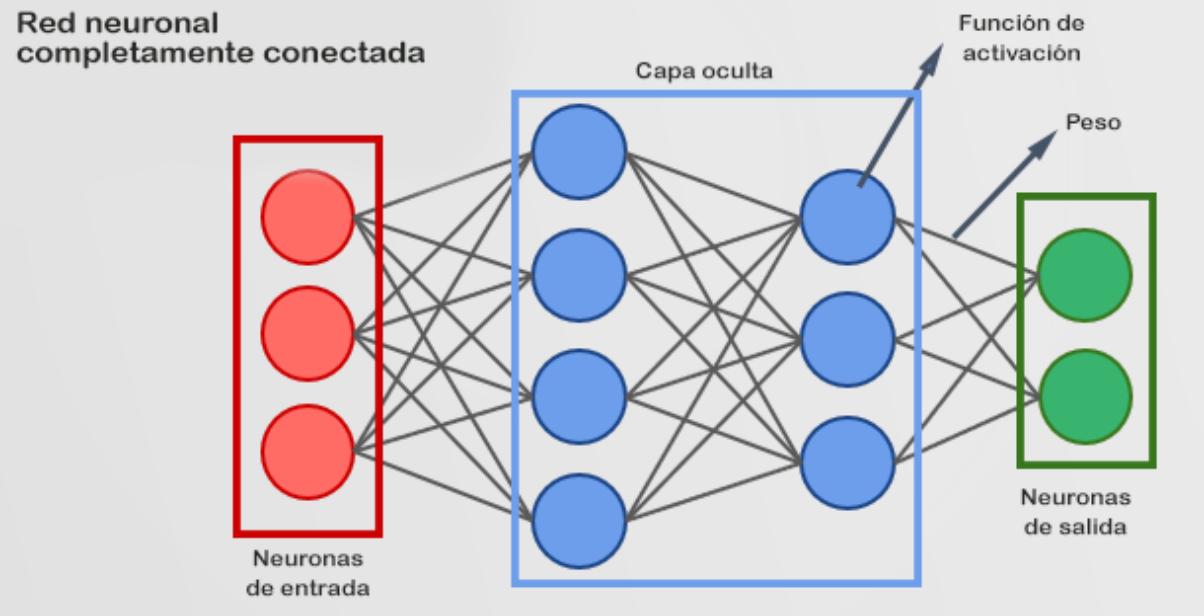


Figura 1: Ejemplo de red neuronal

Hay distintos tipos de redes neuronales según la forma en la que se conectan las neuronas. En esta tesina son de interés especialmente las *Convolutional Neural Networks* (CNN) y las *Recurrent Neural Networks* (RNN), redes neuronales convolucionales y recurrentes respectivamente. Las primeras son útiles para el reconocimiento de patrones en los datos, mientras que las últimas son especialmente buenas en la predicción de datos secuenciales.

Otro tipo de modelo de aprendizaje profundo son los *transformer models* (modelos transformadores), los cúales son siginificativamente más eficientes al entrenar y realizar inferencias que las CNNs y las RNNs gracias al uso de mecanismos de atención, presentados en la publicación '*Attention is all you need*' de Google.

3.3.1 Long Short Term Memory (LSTM)

Lo que caracterizan a las redes neuronales recurrentes son los bucles de retroalimentación que se presentan en la Figura 2. Mientras que cada neurona de entrada en una red neuronal completamente conectada es independiente, en las redes neuronales recurrentes se relacionan entre ellas y se retroalimentan.

Los modelos LSTM son un tipo de RNN que, para evitar que los gradientes se vayan a cero o al infinito muy rápidamente al actualizar los parámetros, usa un algoritmo logístico de 3 puertas:

- Puerta de guardado: Guarda la información relevante del estado actual de la red neuronal.
- Puerta de entrada: Guarda la información que debe ser actualizada en la red neuronal.
- Puerta de salida: Combina la información de las puertas de guardado y entrada, y decide que tanto de esa información usar para seguir utilizando en la red neuronal.

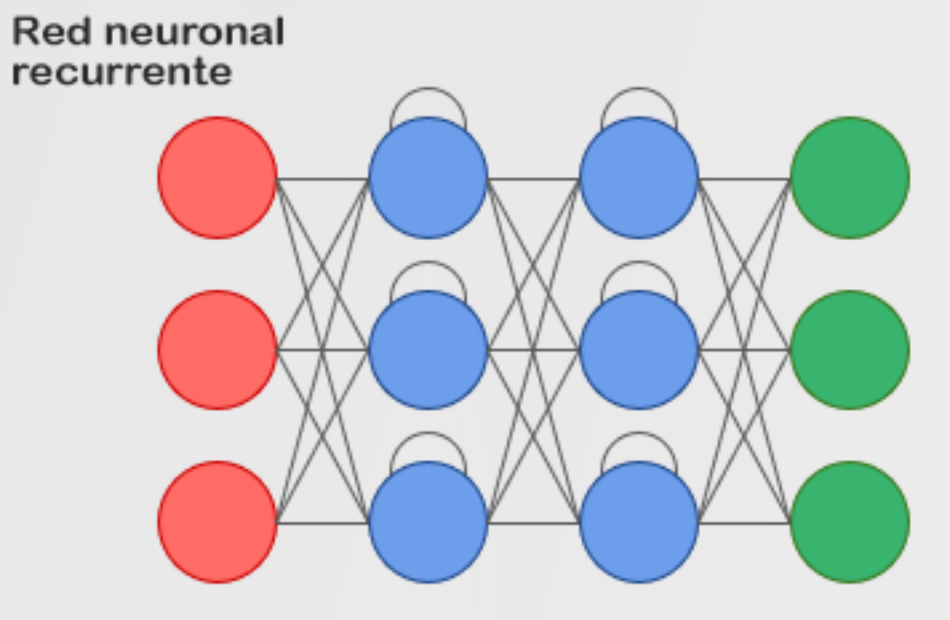


Figura 2: Ejemplo de RNN

3.3.2 TimeGPT

TimeGPT es el primer modelo fundacional pre-entrenado para el pronóstico de series de tiempo que puede producir predicciones en diversas áreas y aplicaciones con gran precisión y sin entrenamiento adicional. Los modelos pre-entrenados constituyen una gran innovación haciendo que el pronóstico de series de tiempo sea más accesible, preciso, tenga menor complejidad computacional y consuma menos tiempo.

TimeGPT es un modelo de tipo transformer con mecanismos de autoatención que no es de código abierto. La autoatención captura dependencias y relaciónes en la secuencias de valores que se alimentan al modelo, logrando poner en contexto a cada observación.

La arquitectura del modelo consiste en una estructura con codificador y decodificador de múltiples capas, cada una con conexiones residuales y normalización de capas. Por último, contiene una capa lineal que mapea la salida del decodificador a la dimensión del pronóstico.

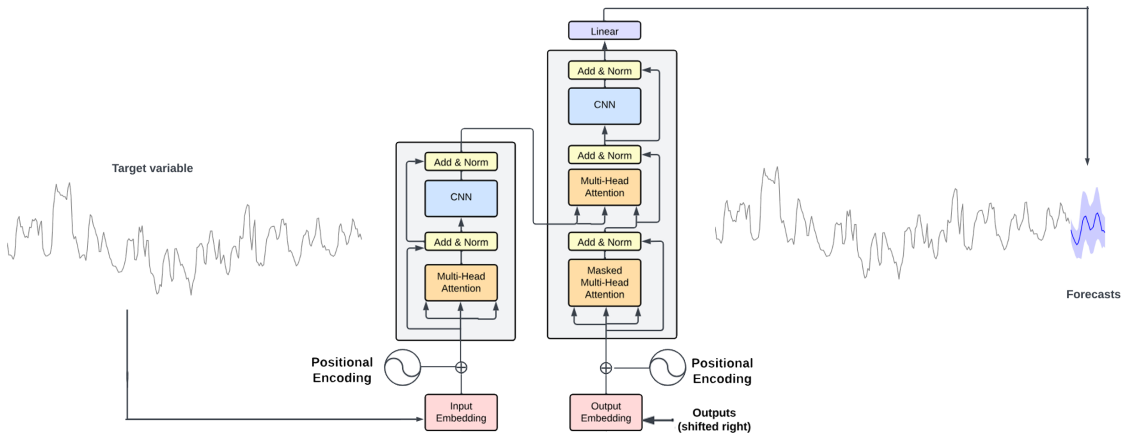


Figura 3: Diagrama de la estructura del modelo

3.3.3 Chronos

Chronos es una familia de modelos fundacionales pre-entrenados para series de tiempo basados en arquitecturas de modelos de lenguaje. Chronos trabaja transformando la serie de tiempo en una secuencia de *tokens* por medio de escalado y cuantificación, para luego entrenar un modelo de lenguaje con estos, usando entropía cruzada. Una vez entrenado el modelo, se generan múltiples pronósticos probabilísticos a partir del contexto historico.

3.4 Métricas de evaluación

Para comparar la calidad de los pronósticos de distintos modelos es necesario contar con algún método más formal e insesgado que simplemente la representación visual. Es por esto que se utilizan distintas medidas del error sobre los valores pronosticados. Para los pronósticos puntuales se utilizará el porcentaje del error absoluto medio, más conocido como MAPE, mientras que para los pronósticos probabilísticos se utilizará el *Interval Score* propuesto por Gneiting y Raftery en '*Strictly Proper Scoring Rules, Prediction, and Estimation*', los mejores pronósticos serán aquellos que tengan los valores más bajos en estas medidas.

4. Aplicación

Se utilizarán varias series provenientes de diferentes contextos, longitudes y características, series que tendrán o no estacionalidad y/o tendencia. Con cada una de estas series se realizará un pronóstico tanto puntual como probabilístico con cada uno de los modelos descriptos anteriormente. En el ajuste de cada modelo también se probarán las distintas opciones de parametrizaje con las que cuenta cada uno, para luego elegir la mejor configuración al realizar el pronóstico final. Con todo esto el esquema de trabajo para cada serie y cada modelo será el siguiente:

1. Presentación de la serie de tiempo con la que se trabajará.
2. Dividir la serie en conjunto de entrenamiento y testeo, según el largo de pronóstico buscado.
3. Apartar un conjunto de datos de validación a partir del conjunto de entrenamiento.
4. Ajustar con el conjunto de entrenamiento modelos con distintas configuraciones, pronosticando el conjunto de validación.
5. Comparar con el conjunto de validación y elegir la mejor configuración para el modelo.
6. Ajustar el modelo con el conjunto de entrenamiento y validación, usando los parámetros elegidos en el paso anterior, buscando pronosticar los valores en el conjunto de entrenamiento.
7. Comparar los resultados con el conjunto de entrenamiento.

Luego se compararán los resultados con el conjunto de entrenamiento entre todos los tipos de modelos y se hablará de las ventajas y desventajas de cada uno.

5. Bibliografía

3Blue1Brown. (2024). Attention in transformers, step-by-step | DL6 . Youtube. https://www.youtube.com/watch?v=eMlx5fFNoYc&t=1204s

3Blue1Brown. (2024). Transformers (how LLMs work) explained visually | DL5 . Youtube. https://www.youtube.com/watch?v=wjZofJX0v4M

Ansari et al. (2024). Chronos: Learning the Language of Time Series. Transactions on Machine Learning Research.

Awan, A. A. (2024). Time Series Forecasting With TimeGPT. Datacamp.

Elhariri, K. (2022). The Transformer Model. Medium.

Gilliland, M., Sglavo, U., & Tashman, L. (2016). Forecast Error Measures: Critical Review and Practical Recommendations. John Wiley & Sons Inc.

Gneiting, T., & Raftery A. E. (2007). Strictly Proper Scoring Rules, Prediction, and Estimation. Journal of the American Statistical Association.

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: principles and practice (3rd ed.). OTexts.

IBM. (s.f.). Explainers. Recuperado el 14 de marzo de 2025 en https://www.ibm.com/think/topics

Kamtziris, G. (2023). Time Series Forecasting with XGBoost and LightGBM: Predicting Energy Consumption. Medium.

Korstanje, J. (2021). Advanced Forecasting with Python. Apress.

Nielsen, A. (2019). Practical Time Series Analysis: Prediction with Statistics and Machine Learning. O'Reilly Media.

Nixtla. (s.f.). About TimeGPT. Recuperado en diciembre de 2024 de https://docs.nixtla.io/docs/getting-started-about_timegpt

Nixtla. (s.f.). LSTM. Recuperado el 9 de abril de 2025 en https://nixtlaverse.nixtla.io/neuralforecast/models.lstm.html#lstm

Shastri, Y. (2024). Attention Mechanism in LLMs: An Intuitive Explanation. Datacamp.

Silberstein, E. (2024). Tracing the Transformer in Diagrams. Medium.

Vaswani et al. (2017). Attention is all you need. Google.