

UNIVERSIDAD DE LOS ANDES
FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS



PROPUESTA DE MODELOS DE INTELIGENCIA ARTIFICIAL
PARA PREDICCION DE TENDENCIA EN ACTIVOS
FINANCIEROS

EDUARDO NICOLAS SINNING QUIROZ

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELÉCTRICO

PROFESOR GUÍA: GUILLERMO ESPINOZA ARMIVO

SANTIAGO, AGOSTO DE 2021

© Eduardo Nicolas Sinning Quiroz 2021

Todos los derechos reservados.

Resumen

El realizar predicciones en el mercado financiero sobre variables macroeconómicas o precios de activos, es de gran complejidad incluso para los algoritmos diseñados para análisis cuantitativos, debido a la naturaleza aleatoria de las variables y por consiguiente en el bajo ratio señal-ruido. Además, la gran cantidad de variables que afectan el precio intrínseco de un activo en la Bolsa de Valores, genera una gran dificultad en la toma de decisiones de inversión, por lo tanto, esta tarea presenta altos costos en tiempo, tanto de ejecución como de aprendizaje.

Investigaciones recientes en el campo de la inteligencia artificial (AI) tanto en redes neuronales profundas (Deep Learning) como en procesamiento de lenguaje natural (NLP) han obtenido resultados esperanzadores en la reducción de estos altos costos. Permitiendo de esta manera, extraer información de noticias en forma autónoma, para luego realizar predicciones y así, acelerar la generación de estrategias cuantitativas en inversiones.

En este trabajo de título, se evalúa la clasificación de sentimientos extraídos desde noticias financieras, usando modelos basados en transformadores. En particular, usamos Representación de Codificador Bidireccional de Transformadores del inglés Bidirectional Encoder Representations from Transformers (BERT) para luego incorporar dicha información como entrada a modelos predictivos basados en Long-Short Term Memory (LSTM), y así, generar proyecciones del comportamiento futuro en el precio de cierre de activos financieros. Del mismo modo, se realizan estudios con modelos generativos adversarios, los cuales permiten entregar información de la distribución de probabilidades intrínseca al comportamiento del activo.

Para evaluar los beneficios de ambas predicciones, se presenta una simulación mediante aprendizaje reforzado en un entorno de compra y venta de activos. En ésta, se compara el desempeño de un agente con ambas predicciones y sin información futura. Los resultados tanto a nivel cualitativo de las predicciones como en el experimento final, permiten concluir que ambos modelos predictivos presentan información útil, respecto de la toma de decisiones financieras.

Agradecimientos

Le agradezco a todo aquel que tuvo la intención de enseñarme algo en este largo camino.

Este trabajo esta dedicado a mi madre que insistio hasta el final que me tenia que titular.

Índice General

Resumen	III
Agradecimientos	IV
1. Introducción	1
1.1. Motivación y Contexto	1
1.2. Objetivos	4
1.2.1. Objetivos específicos	4
1.3. Estructura de la Memoria	4
2. Marco Teórico	6
2.1. Redes Neuronales Artificiales	6
2.1.1. Neurona	6
2.1.2. Entrenamiento	7
2.1.3. Redes Neuronales Recurrentes	9
2.1.4. Redes Neuronales Generativas Adversarias	11
2.2. Preprocesamiento de Datos	15
2.3. Aprendizaje Reforzado	16
2.4. Clasificación de Sentimiento en Noticias Financieras	20
2.4.1. Entrenamiento adversario para un clasificador de sentimientos	21
3. Metodología	23
4. Clasificación de Sentimientos Financiero	26
4.1. Datos	27
4.2. Entrenamiento Supervisado	28
4.3. Entrenamiento Léxico Adversario	28
4.4. Análisis de Evaluación Comparativo	30
5. Preprocesamiento de Datos	32
5.1. Recopilación de Datos	32

5.1.1.	Activos Relacionados	33
5.1.2.	Indicadores Tecnicos	34
5.1.3.	Modelo Autorregresivo Integrado de Media Móvil	35
5.1.4.	Componentes de Fourier	36
5.1.5.	Análisis de Sentimiento en Fuentes Financieras	37
5.2.	Categorización de Variables según su Nivel de Importancia	38
5.3.	Características de Datos para Modelos Predictivos	43
6.	Modelos Predictivos para Activos Financieros	44
6.1.	Modelo con Arquitectura basada en Redes LSTM	44
6.2.	Modelo con Arquitectura basada en Redes Generativas Adversarias .	49
6.2.1.	Experimento 1	50
6.2.2.	Experimento 2	50
6.2.3.	Experimento 3	56
6.3.	Evaluación Comparativa entre Proyecciones basadas en LSTM y arquitectura TimeGAN	57
6.3.1.	Comparación Cualitativo de modelos predictivos	58
6.3.2.	Experimento en Entorno Financiero	60
7.	Conclusiones	63
7.1.	Discusión	63
7.2.	Futuras Investigaciones	65
Bibliografía		66
Anexos		69
A. Glosario		69
B. Redes Neuronales Recurrentes		71
B.1.	Componentes Fundamentales	71
B.1.1.	Dificultades de entrenamiento	80
B.1.2.	Long Short-Term Memory	84
C. Transformadores		87
C.1.	Características de Transformadores	87
C.2.	Arquitectura de codificador	89
C.3.	Arquitectura del decodificador	91

D. Resultados	93
D.1. Experimento 1: Graficos	93
D.2. Experimento 3: Graficos	95

Listo de Ilustraciones

2.1. Modelo no lineal de una neurona.	7
2.2. Diagrama del proceso de retropropagación.	8
2.3. Neurona recurrente, equivalente a un conjunto de neuronas normales concatenadas.	9
2.4. Neurona LSTM, adaptada de una RNN canónica.	10
2.5. Representación análoga de la estructura generativa adversaria	12
2.6. Diagrama estructural de TimeGAN	13
2.7. Proceso de Aprendizaje Reforzado	17
2.8. Implementación de Aprendizaje Reforzado Profundo para Optimización de Hiperparametros	18
2.9. Implementación de Aprendizaje Reforzado Profundo para entorno financiero	19
2.10. La descripción general de los pasos para capacitar a FinBERT	20
2.11. Entrenamiento Léxico Adversario,(”Xu, Zhao, Yan, Zeng, Liang & Sun, 2019)	22
3.1. Diagrama de bloque de las diferentes etapas de la Metodología	23
4.1. Esquema metodológico en la implementación del clasificador de sentimiento financiero	26
5.1. Estructura relacional de la base de datos	33
5.2. Gráficos de Indicadores Técnicos para el precio de cierre de 'INTC' . .	35
5.3. Predicción del precio de cierre del Modelo ARIMA para Intel Corporation (“ INTC ”)	36
5.4. Componentes de Fourier del precio de cierre para Intel Corporation (“ INTC ”)	37
5.5. Diagrama del Análisis de Noticias Financieras.	38
5.6. 10 Acciones del Dow Jones de mayor importancia para “ INTC ”.	40

5.7.	Los 10 Índices Globales de mayor Importancia para “ INTC ”	41
5.8.	Las 10 Monedas Globales de mayor factor de relevancia para 'INTC'	41
5.9.	10 Indicadores Técnicos con mayor relevancia para “ INTC ”	42
5.10.	10 variables contextuales con mayor relevancia para “ INTC ”	43
6.1.	Funcion de Accuracy (Diferencia Porcentual) utilizado como métrica cuantitativa en todos los modelos predictivos.	45
6.2.	Diagrama de Bloques de la búsqueda de Hiperparametros para modelos basados en LSTM	46
6.3.	Proceso de Optimización de Hiperparametros en arquitectura LSTM Tipo 1.	47
6.4.	Proceso de Optimización de Hiperparametros en arquitectura LSTM Tipo 2.	47
6.5.	Gráfico Comparativo en modelos basados en LSTM ('INTC'). Ventana evaluada : [2021-02- 22/2021-02- 26]. Ventana predictiva : [2021-03-01/2021-03-05].	48
6.6.	Gráfico Comparativo en modelos basados en LSTM ('INTC'). Ventana evaluada: [2021-03- 05/2021-03- 10]. Ventana predictiva : [2021-03-11/2021-03-15].	49
6.7.	Gráfico Comparativo en modelos basados en LSTM ('INTC'). Ventana evaluada : [2021-03- 15/2021-03- 19]. Ventana predictiva : [2021-03-22/2021-03-26].	49
6.8.	Diagrama de Bloques de la búsqueda de hiperparametros para los modelos basados en TimeGAN	51
6.9.	Diagrama metodológico del entrenamiento realizado en los modelos basados en TimeGAN.	52
6.10.	Experimento 2. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]	53
6.11.	Experimento 2. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]	53
6.12.	Experimento 2. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-9/2021-03-15]	54

6.13. Experimento 2. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-9/2021-03-15]	54
6.14. Experimento 2. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]	55
6.15. Experimento 2. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]	55
6.16. Experimento 2. Gráfico Comparativo Promedio ('INTC') con todo el conjunto de datos de evaluación . Conjunto de evaluación : [2021-02-23/2021-06-15]	56
6.17. Experimento Comparativo. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 01/2021-03- 05]. Ventana predictiva : [2021-03-08/2021-03-12]	58
6.18. Experimento Comparativo. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-09/2021-03-15]	59
6.19. Experimento Comparativo. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 03/2021-03- 09]. Ventana predictiva : [2021-03-10/2021-04-16]	59
6.20. Experimento Comparativo. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [[2021-03- 04/2021-03- 10]. Ventana predictiva : [2021-03-11/2021-04-17]]	60
6.21. Experimento Comparativo en entorno Financiero (DQL)	61
 C.1. Diagrama de transformador ilustrada en (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser & Polosukhin, 2017)	88
C.2. Ejemplo ilustrativo de la estructura del encoder con D=512 y S=4 que representa la duración máxima de la sentencia según el lote	89
C.3. Atención de productos escalados	90
C.4. La atención de múltiples encabezados consta de varias capas de atención, ilustración C.3, que se ejecutan en paralelo	91
 D.1. Experimento 1. Gráfico Comparativo en TimeGAN original ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]	93

D.2. Experimento 1. Gráfico Comparativo en TimeGAN original ('INTC'). Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-9/2021-03-15]	94
D.3. Experimento 1. Gráfico Comparativo en TimeGAN original ('INTC'). Ventana de Datos : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]	94
D.4. Experimento 3. Gráfico Comparativo Completo ('INTC'). Ventana eva- luada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021- 03-08]	95
D.5. Experimento 3. Gráficos Comparativos Promedio ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03- 02/2021-03-08]	95
D.6. Experimento 3. Gráfico Comparativo Completo ('INTC'). Ventana eva- luada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021- 03-22]	96
D.7. Experimento 3. Gráfico Comparativo Promedio ('INTC'). Ventana eva- luada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021- 03-22]	96
D.8. Experimento 3. Gráfico Comparativo Completo ('INTC'). Ventana eva- luada : [2021-03- 23/2021-03- 29]. Ventana predictiva : [2021-03-30/2021- 04-05]	97
D.9. Experimento 3. Gráfico Comparativo Promedio ('INTC'). Ventana eva- luada : [[2021-03- 23/2021-03- 29]. Ventana predictiva : [2021-03-30/2021- 04-05]]	97
D.10.Experimento 3. Gráfico Comparativo Promedio ('INTC') con todo el conjunto de datos de evaluación . Conjunto de evaluación : [2021-02- 23/2021-06-15]	98

Lista de Tablas

4.1.	Distribución de etiquetas en banco de frases para 4 subconjuntos formados en base a la fuerza del acuerdo mayoritario. Cada oración tiene 5 a 8 anotaciones superpuestas, que se han utilizado para determinar el grado de acuerdo.	27
4.2.	Resultados de Entrenamientos Léxico Adversarios	29
4.3.	Resultados de entrenamientos para clasificador de sentimiento de noticias financieras	30
5.1.	Tipo de monedas Globales evaluadas en esta investigación	34
5.2.	Definición de indicadores técnicos	34
5.3.	Resultados de Optimización de Hiperparametros algoritmo XGBoost	39
6.1.	Tabla de parámetros a optimizar en arquitecturas basadas en LSTM.	45
6.2.	Resultados de parámetros óptimos en modelos basados en LSTM según tipo de dato de entrada.	48
6.3.	Experimento 2.Tabla de parámetros óptimos y rango en el proceso de optimización para TimeGAN.	51
6.4.	Experimento 3.Tabla de parámetros óptimos y rango en el proceso de optimización para TimeGAN	57
6.5.	Experimento Comparativo.Tabla de evaluación cuantitativa para 'INTC'. .	57
6.6.	Resultados de experimento en entorno financiero	61

Capítulo 1

Introducción

1.1. Motivación y Contexto

Es de conocimiento general que realizar predicciones en el mercado financiero es una tarea de alta complejidad. De tal magnitud es esa dificultad, que existen corrientes de pensamiento que plantean la imposibilidad de esta tarea. La hipótesis de los mercados eficientes (Fama, 1965), afirma que un mercado de valores es: “Informacionalmente eficiente cuando la competencia entre los distintos participantes que intervienen en el mismo conduce a una situación de equilibrio en la que el precio de mercado de un activo constituye una buena estimación de su precio teórico o intrínseco”. Expresado de otra forma, los precios de las acciones que se negocian en un mercado financiero eficiente, reflejan toda la información existente y se ajustan total y rápidamente a los nuevos datos que puedan surgir.

Sin embargo el ultimo tiempo, el cuestionamiento de esta hipótesis ha provocado la aparición de economistas, matemáticos y especuladores que no avalan en plenitud la propuesta del mercado eficiente. Ellos piensan que hay razones poderosas para que existan ineficiencias tales, como la lenta difusión de la información, el poder desigual de los distintos participantes en los mercados y la evidencia empírica de inversores profesionales quienes logran altos rendimientos mediante el análisis técnico y fundamental (Malkiel, 2003). Esta nueva corriente, sumada a las últimas tecnologías de procesamiento de información y detección de patrones permite dar inicio a esta investigación.

En ciencias de la computación, existe un conjunto de algoritmos llamado aprendizaje automático los cuales permiten a las máquinas aprender. Aprender en este contexto quiere decir identificar patrones complejos en millones de datos. Por lo tanto, la máquina que realmente aprende, es un algoritmo que revisa los datos y es capaz

de predecir comportamientos futuros o tomar decisiones.

Este tipo de algoritmos presenta 3 etapas dentro de las que analiza distintos datos, es por ello que el conjunto total de datos necesita ser dividido en 3 subconjuntos. Estas etapas se denominan de **entrenamiento, testeо y evaluación**.

Los algoritmos de aprendizaje automático se dividen en tres categorías, según la forma en cómo aprende el modelo (Mohammed, Khan & Bashier, 2016).

En primer lugar se encuentran aquellos que cuentan con un aprendizaje previo basado en un sistema de etiquetas asociados a datos de entradas que les permiten tomar decisiones, clasificar o realizar predicciones. Básicamente el programador le enseña a la máquina cuales son los resultados deseados, es por ello que se denominan **algoritmos de aprendizaje supervisado**.

En segundo lugar estan aquellos que se enfrentan al caos de los datos con el objetivo de encontrar patrones que permitan organizarlos de alguna manera. Como no presentan un conocimiento previo, ni indicaciones claras por parte del programador, se denominan **algoritmos de aprendizaje no supervisado**.

En tercer lugar están aquellos algoritmos que representan un aprendizaje conductual. Mediante este tipo de algoritmos el sistema aprende a través de la prueba y error reforzando aquellas decisiones exitosas, es por ello que se denominan **algoritmos de aprendizaje reforzado** .

Existen diversos algoritmos que permiten a las computadoras realizar estos tipo de aprendizaje, siendo las redes neuronales artificiales las que simulan de mejor manera al cerebro humano (Olabe, 2008). Las redes neuronales artificiales, son algoritmos que han sido motivado desde sus inicios a replicar la forma en que el cerebro humano computa. El cerebro, en términos generales es una computadora, es decir es un sistema de procesamiento de información, altamente compleja, no lineal y paralela. Éste tiene la capacidad de organizar sus componentes estructurales, conocidos como neuronas, para realizar ciertos cálculos y tareas. Un cerebro tiene una estructura y la capacidad de construir sus propias reglas de comportamiento a través de lo que generalmente llamamos “experiencia”, mediante conexiones sinápticas que dotan al individuo de conocimiento. Este principio se busca replicar con las redes neuronales artificiales.

A partir de un estudio general de herramientas de inteligencia Artificial, utilizadas en el área financiera (Jiang, 2020), se identifican tres aspectos relevantes para esta investigacion. En primer lugar, se encuentra la **clasificación automatica de sentimiento**, en artículos financieros mediante algoritmos de aprendizaje supervisado, lo

que permite extraer información útil para análisis fundamental, de un considerable número de artículos sin el costo de tiempo que eso implica.

En segundo lugar, están las diferentes arquitecturas utilizadas para construir **predicciones futuras**.

Finalmente, la **toma de decisiones en forma automática** mediante algoritmos de aprendizaje reforzado permite obtener resultados más prometedores y en un menor tiempo . Estos tres aspectos, son los que se abordan en esta investigación.

1.2. Objetivos

El objetivo principal de este trabajo es usar diferentes herramientas de vanguardia en inteligencia artificial y procesamiento de datos en distintos escenarios experimentales con el propósito de evidenciar información útil y relevante en la toma de decisiones en determinadas actividades financieras.

1.2.1. Objetivos específicos

De acuerdo con lo anterior, los objetivos específicos de este trabajo son los siguientes:

- Identificar los factores que afectan en el precio de un activo y obtener valores de importancia según el tipo de activo.
- Implementar un clasificación de sentimientos financieros, con un nivel de robustez aceptable.
- Identificar patrones a partir de modelos ampliamente utilizados en series de tiempo, LSTM, con resultados predictivos aceptables.
- Implementar pruebas mediante modelos generativos para series de tiempo con resultados cualitativos aceptables.
- Diseñar un experimento comparativo en un entorno financiero que permita dilucidar un comportamiento real de la información entregada por los modelos predictivos.

1.3. Estructura de la Memoria

Este documento está organizado de la siguiente manera:

En el **Capítulo 2** se presenta el marco teórico; en él se describen los conceptos fundamentales necesarios para la comprensión de este documento. En esta sección se aborda las herramientas de vanguardia utilizadas en este trabajo.

En el **Capítulo 3**, se expone la metodología realizada pa cumplir con los objetivos señalados anteriormente.

En el **Capítulo 4**, se describe el procedimiento utilizada para la implementacion de un clasificacion de sentimiento financiero, con un nivel de robustez aceptable.

En el **Capítulo 5**, se presentan las herramientas utilizadas para preparar los datos que se utilizaran en los modelos predictivos.

En el **Capítulo 6**, se presenta la metodología realizada en los modelos predictivos.

Finalmente, en el **Capítulo 7**, se presentan las conclusiones generales; en estas, se determina la utilidad de los modelos implementados en esta investigación, pues con estos se obtienen mejores resultados en el experimento en comparación con los datos reales, . También se presentan posibles investigaciones futuras.

Capítulo 2

Marco Teórico

En este capítulo se presentará una breve definición de los términos fundamentales necesarios para comprender el contenido de este documento.

En primer lugar, se describen a grandes rasgos lo que son las redes neuronales y sus características para luego describir aquellos arquitecturas utilizados para modelar el comportamiento temporal de los activos.

En segundo lugar, se describen tanto el tipo de dato como el preprocesado de estos para los modelos generativos adversarios .

En tercer lugar se describe que es el aprendizaje reforzado junto con la forma en que fueron implementados en este trabajo.

Finalmente se presentan las herramientas de inteligencia artificial utilizadas para la clasificación de sentimiento de noticias financieras.

2.1. Redes Neuronales Artificiales

Una Red Neuronal Artificial es una estructura organizada en capas construidas a partir de unidades de procesamiento de información simples , llamadas neuronas, que mediante un algoritmo de aprendizaje , se estructuran las conexiones o pesos sinápticos entre neuronas permitiendo recopilar conocimiento a partir de la información que le fue entregada (Haykin, 2009).

2.1.1. Neurona

Una neurona es la unidad de procesamiento de información fundamental de toda red neuronal artificial. El diagrama de bloques presentado en la Ilustración 2.1 describe los 3 componentes claves del modelo neuronal, que constituye la base para diseñar

una gran familia de redes neuronales.

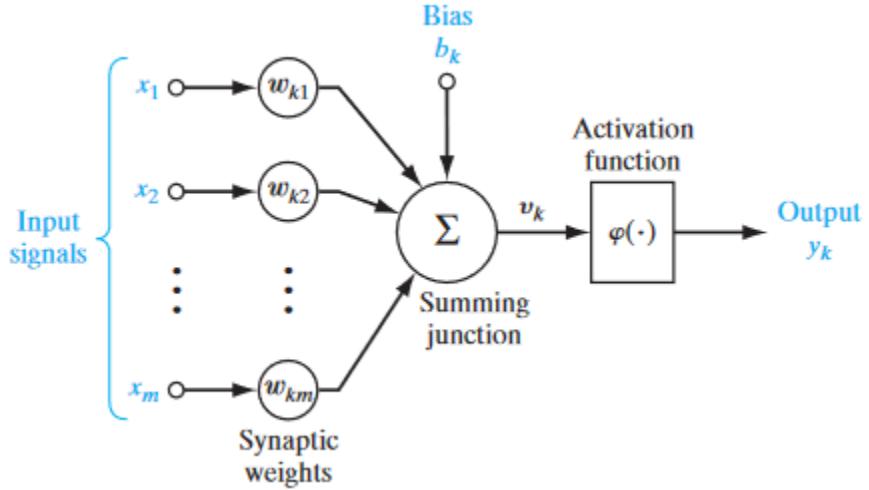


Ilustración 2.1: Modelo no lineal de una neurona.

En primer lugar cada neurona presenta un conjunto sináptico, o enlaces de conexiones, cada uno de los cuales se caracteriza por un peso . Específicamente, una señal x_j en la entrada de la sinapsis j conectada a la neurona k se multiplica por el peso sináptico w_{kj} . Es importante tomar nota de la forma en que se escriben los subíndices del peso sináptico w_{kj} . El primer subíndice se refiere a la neurona en cuestión, y el segundo subíndice se refiere al extremo de entrada de la sinapsis al que se refiere el peso. A diferencia del peso de una sinapsis en el cerebro, el peso sináptico de una neurona artificial puede estar en un rango que incluye valores tanto negativos como positivos. En segundo lugar presenta un sumador de señales de entrada, previamente ponderadas por sus respectivos pesos sinápticos formando así una combinación lineal. El modelo neuronal también incluye un sesgo aplicado externamente, denotado por b_k , el cual permite aumentar o disminuir la entrada neta de la función de activación, dependiendo de si es positiva o negativa, respectivamente. Finalmente este resultado es introducido como valor de entrada a una función de activación para limitar la amplitud de la salida. La función de activación también se conoce como función de aplastamiento, ya que aplasta (limita) el rango de amplitud permisible de la señal de salida a un valor finito.

2.1.2. Entrenamiento

Entrenar una red neuronal consiste en ajustar cada uno de los pesos de todas las neuronas que forman parte de la red neuronal, para que las respuestas de la capa de

salida se ajusten lo más posible a los datos que conocemos. Los pesos asociados a las neuronas pueden actualizarse de muchas maneras para optimizar la red, pero un manera especial, que revolucionó el aprendizaje supervisado mediante redes neuronales, es el de backpropagation o retropropagación (Rumelhart, Hinton & Williams, 1986). Esta técnica permite actualizar los pesos de toda la red en función de lo “bien o mal” que se ha comportado la red al ejecutarse sobre una muestra para la que se conoce la salida deseada. El procedimiento para distinguir lo assertivo de los resultados consiste en minimizar una función de coste (como por ejemplo un error de aproximación, un error cuadrático medio, etc) ¹ a partir de un algoritmo denominado descenso de gradiente.

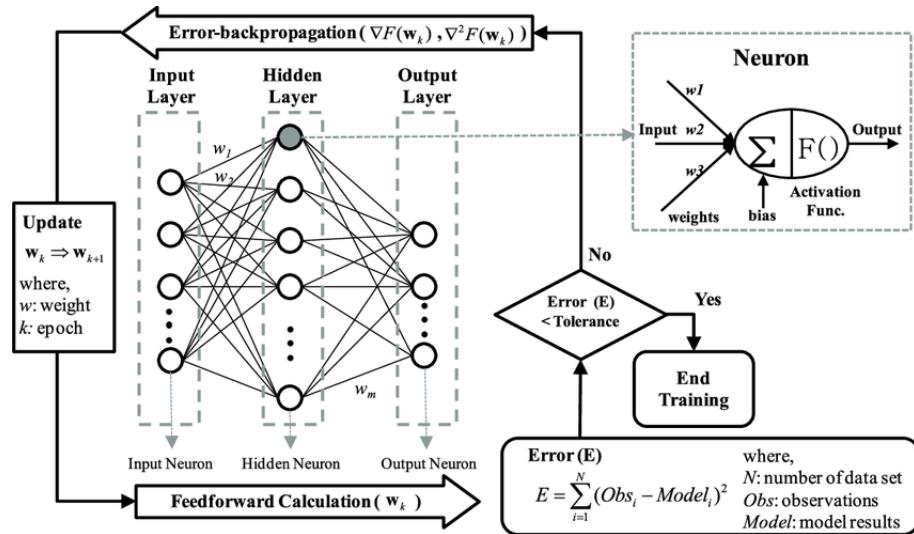


Ilustración 2.2: Diagrama del proceso de retropropagación.

A grandes rasgos, este algoritmo iterativo consta de los siguientes pasos:

- Se calcula la salida de la red a partir de los valores de entrada de los datos de entrenamiento.
- Se compara la salida con la salida esperada mediante la función de coste, denotada C , obteniendo una señal del error.
- Calcular las derivadas parciales del error con respecto a los pesos $\frac{\partial C}{\partial w}$ capa por capa desde la salida hasta el comienzo de la red ².
- Ajusta los valores de los pesos de cada neurona de la red para reducir el error según su porción calculada.

¹La función de coste es dependiente del problema.

²La matriz de derivadas parciales se denomina el gradiente.

- e) Se repite nuevamente todo el proceso hasta llegar al mínimo global.

2.1.3. Redes Neuronales Recurrentes

Una red neuronal recurrente (RNN) es un tipo de red neuronal artificial que utiliza datos secuenciales o datos de series de tiempo. Se distinguen por su “memoria”, ya que toman información de entradas anteriores para influir en la entrada y salida actuales. Mientras que las redes neuronales tradicionales asumen que las entradas y salidas son independientes entre sí, la salida de las redes neuronales recurrentes depende de los elementos anteriores dentro de la secuencia (Sherstinsky, 2020).

En el modelo de neurona antes visto, Ilustración 2.1, la función de activación solo actúa en una dirección, hacia delante, desde la capa de entrada hacia la capa de salida, es decir, que no recuerdan valores previos. Lo que distingue a una neurona recurrente es que incluye conexiones que apuntan “hacia atrás”, una especie de retroalimentación entre las neuronas dentro de la capa. Cada neurona recurrente (Ilustración 2.3³) tienen dos conjuntos de parámetros, uno que lo aplica a la entrada de datos que recibe de la capa anterior (W_x) y otro conjunto que lo aplica a la entrada de datos correspondiente al vector salida del instante anterior (W_r)⁴.

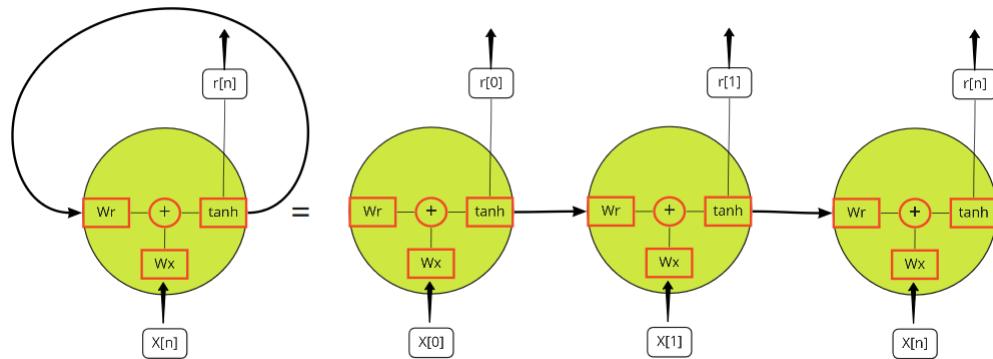


Ilustración 2.3: Neurona recurrente, equivalente a un conjunto de neuronas normales concatenadas.

³Representación grafica de ecuacion B.31

⁴Si el lector desea profundizar diríjase al Anexo B.1 donde se encuentran los fundamentos matemáticos que justifican este tipo de neurona

Por lo tanto una capa de neuronas recurrentes se puede implementar de tal manera que, en cada instante de tiempo, cada neurona recibe dos entradas, la entrada correspondiente de la capa anterior y a su vez la salida del instante anterior de la misma capa.

Los sistemas RNN son problemáticos en la práctica, ya que durante el entrenamiento sufren problemas conocidos como desaparición del gradiente o gradientes explosivos [(Hochreiter & Schmidhuber, 1997), (Hochreiter, Bengio, Frasconi & Schmidhuber, 2001), (Pascanu, Mikolov & Bengio, 2013)]. Estas dificultades se vuelven pronunciadas cuando las dependencias en la subsecuencia de destino abarcan un gran número de muestras, esto requiere que la ventana del modelo RNN deba ser proporcionalmente amplia para capturar estas dependencias de largo alcance ⁵.

La neurona Long Short-Term Memory (LSTM) se inventó con el objetivo de abordar el problema de los gradientes que se desvanecen. La idea clave en el diseño de LSTM fue incorporar controles no lineales, que se pueden capacitar para garantizar que el gradiente de la función objetivo no desaparezca (Hochreiter & Schmidhuber, 1997). Básicamente la incorporación de 3 puertas determinan si se permite o no una nueva entrada ($\vec{G}_{cu}[n]$), se elimina la información porque no es importante ($\vec{G}_{cs}[n]$) o se deja que afecte a la salida en el paso de tiempo actual ($\vec{G}_{cr}[n]$) ⁶.

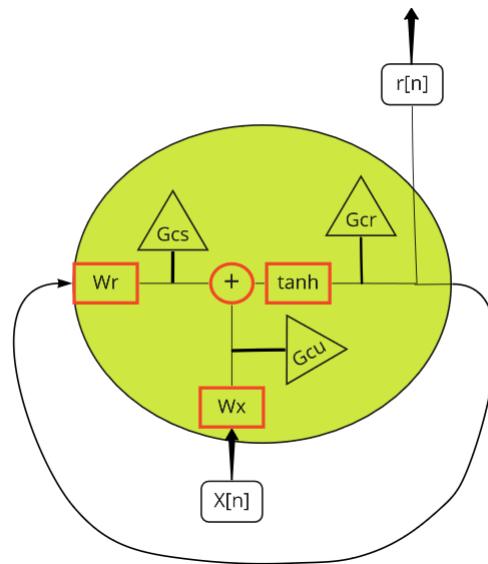


Ilustración 2.4: Neurona LSTM, adaptada de una RNN canónica.

⁵La demostración de este fenómeno está descrito en Anexo B.1.1.

⁶Si el lector desea profundizar diríjase al Anexo B.1.2

Estas puertas son análogas a una forma sigmoide, lo que significa que van de 0 a 1⁷. Esto permite incorporarlas matemáticamente al proceso de Retropropagación a través del tiempo [(Werbos, 1990), (Werbos, 1988)].

Como ya hemos comentado, los problemas de los gradientes que se desvanecen se resuelven a través de LSTM porque mantiene los gradientes lo suficientemente empinados y, por lo tanto, el entrenamiento es relativamente corto y de precisión alta.

2.1.4. Redes Neuronales Generativas Adversarias

Esta red neuronal artificial presenta una estructura de redes adversas formada por dos modelos (Ian J. Goodfellow, 2014). Por un lado se encuentra el modelo generativo, que busca generar muestras nuevas. Por otro lado se encuentra su adversario llamado modelo discriminativo, que aprende a discriminar si las muestras son provenientes del generador o son muestras reales. Estos dos modelos buscan optimizar cosas opuestas, pero su aprendizaje es en conjunto cuyo objetivo es alcanzar el equilibrio de Nash en un juego.

Un ejemplo clásico para explicar el comportamiento de este tipo de red es usando como análogo al generador un equipo de falsificadores que tratan de producir una moneda falsa y usarla sin detección, mientras que el modelo discriminativo es el análogo a la policía, el cual trata de detectar la moneda falsa. La competencia en este juego impulsa a ambos equipos a mejorar sus métodos hasta que las falsificaciones sean indistinguibles con las reales.

El entrenamiento de este tipo de modelos se define como semi-supervisado, ya que el discriminador realiza un pre-entrenamiento supervisado donde aprende a clasificar los datos reales para luego interrelacionarse con el generador en un aprendizaje no supervisado. Dándole así aquella característica generativa deseada.

⁷Representación matemática descrita en Anexo Ecuación B.60.

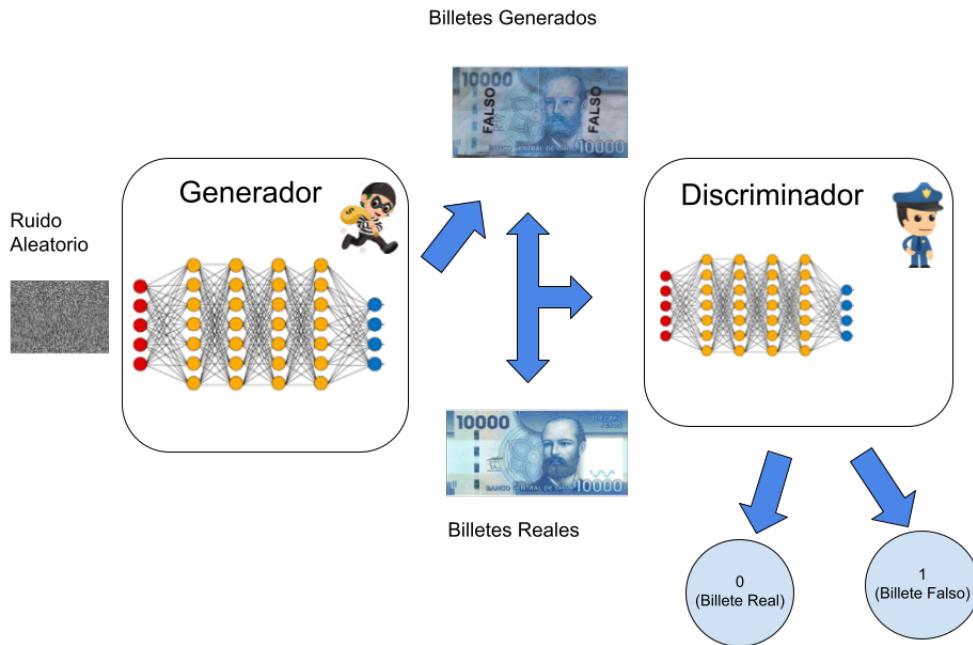


Ilustración 2.5: Representación análoga de la estructura generativa adversaria .

Recientemente han existido diversos avances en propuestas para adaptar la GAN estándar en series de tiempo⁸. Particularmente el framework TimeGAN (Yoon, Jarrett & van der Schaar, 2019), propone un modelo entrenado explicitamente para preservar la dinámica temporal.

Primero, adicionando a la función de coste adversaria (no supervisada) tanto en las secuencias reales como sintéticas, se introduce una función de coste supervisada que usa las secuencias reales como supervisión. Con esto alienta explícitamente al modelo a capturar la distribución condicional paso a paso contenida en los datos. De esta manera es posible extraer más información en los datos de entrenamiento que simplemente determinar qué dato es real o ficticio, permitiéndole a la red aprender expresamente la dinámica de transición de secuencias reales. En segundo lugar se introduce una red embebida para proporcionar un mapeo reversible entre los datos en un espacio característico mediante una representación latente, reduciendo la alta dimensionalidad del espacio de aprendizaje adversario. Con esto se capitaliza el hecho

⁸En particular para predicciones de activos se encuentran (Romero, 2019), pero no son capaces de preservar la dinámica temporal.

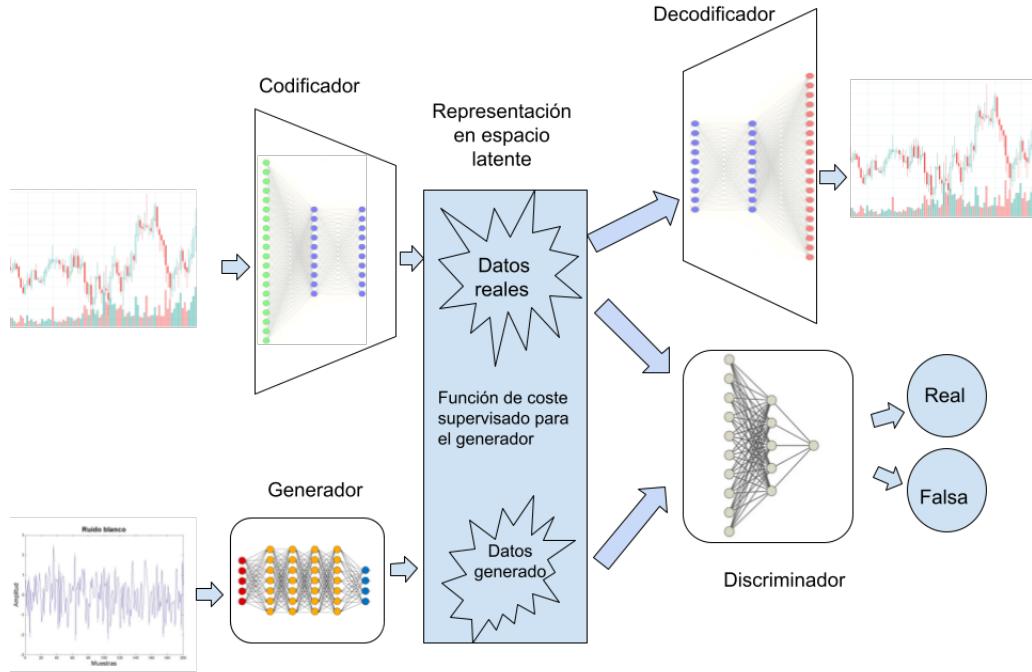


Ilustración 2.6: Diagrama estructural de TimeGAN .

de que la dinámica temporal de incluso sistemas complejos a menudo es impulsada por factores de variación de dimensiones menores.

Es importante destacar que la función de coste supervisado se minimiza mediante el entrenamiento conjunto de las redes embebidas y generativa, de modo que el espacio latente no solo sirve para promover la eficiencia de los parámetros, sino que está especialmente acondicionado para facilitar al generador en el aprendizaje de las relaciones temporales.

La idea clave es que los componentes de codificación automática (en su conjunto como un Autoencoder) se entrena junto con los componentes adversarios (en su conjunto como una GAN estándar), de modo que la TimeGAN aprende simultáneamente a codificar características, generar representaciones e iterar en el tiempo.

El codificador proporciona el espacio latente, la red adversaria opera dentro de este espacio y la dinámica de los datos tanto reales como sintéticos se sincronizan a través de una función de coste supervisada.

Este framework fue probado en el paper con diferentes datos, dentro de los cuales se encuentra la generación de secuencias temporales en el precio de 'APPLE.INC' donde lo que se busca es generar valores realistas que se muevan con la dinámica temporal

aprendida en el entrenamiento a partir de un vector aleatorio como dato de entrada, por ende el objetivo propuesto es la generación de nuevas secuencias a partir de una componente aleatoria, sin condiciones externas.

Como el foco de esta memoria es obtener información útil con respecto a los que podría pasar en el futuro. Se realizan dos experimentos donde los datos de entrada al generador no son valores aleatorios, sino que valores reales de la ventana previa para proporcionar los valores de la próxima ventana cuya cercanía a los valores reales será evaluado tanto por el supervisor como por el discriminador.

Por otro lado, la configuración de cada una de las redes internas propuestas en el documento original están construidas mediante RNN (LSTM) pero estas configuraciones son libres de arquitectura y muchas de las elecciones en el modelo original fueron para simplificar el estudio. Por ende en un tercer experimento se realizar cambios a nivel interno. Para ello se presenta un clasificador basado en redes convolucionales como discriminador. Usualmente las redes convolucionales se usan para trabajos relacionados a imágenes (clasificación, extracción de contexto, etc.), siendo muy poderosas para extraer características de características. Por ejemplo, en una imagen de un perro, la primera capa convolucional detectará bordes, la segunda comenzará a detectar círculos y la tercera detectará una nariz. En nuestro caso, los puntos de datos forman pequeñas tendencias, las pequeñas tendencias forman una más grandes, y estas a su vez forman patrones. La capacidad de las redes convolucionales para detectar características se puede utilizar para extraer información sobre patrones en los movimientos del precio de cierre del activo. Otra razón para usar este tipo de red es que funcionan bien con datos espaciales, lo que significa que los puntos de datos que están más cerca entre sí tienen una relación mayor comparado con los que se encuentran más dispersos. Esto contribuye en el análisis de series de tiempo, donde la relación presente entre dos días cercanos debe de ser mayor entre aquellos más alejados.

En la estructura interna del discriminador se implementa una normalización a nivel de lote. Normalizar los datos permite que las distancias de los datos se distribuya entre 0 y 1, lo cual ayuda a la red neuronal a trabajar mejor. Si bien todos los datos, tanto de entrenamiento como de evaluación están normalizados, esto solo beneficia a la capa de entrada pero conforme los datos pasan por otras capas ocultas esta normalización se va perdiendo y si tenemos una red neuronal con muchas capas podemos tener problemas con el entrenamiento. El método implementado normaliza los datos antes de que pasen por la función de activación en cada capa de la red neuronal, de esta manera siempre tendremos los datos normalizados. La función de activación presente en esta arquitectura entre las capas convolucionales fue la unidad lineal rectificada con

fugas ('LeakyRelu' o ReLU con fugas). Esta es un tipo de función de activación basada en la unidad lineal rectificada (ReLU), pero tiene una pendiente pequeña para valores negativos en lugar de una pendiente plana. El coeficiente de pendiente se determina antes del entrenamiento, es decir, no se aprende durante el entrenamiento. Este tipo de función de activación es popular en tareas en las que podemos sufrir de gradientes escasos, muy comúnmente en entrenamientos de redes generativas adversarias.

Establecer la tasa de aprendizaje para casi todos los optimizadores (como SGD, Adam o RMSProp) es de vital importancia al entrenar redes neuronales porque controla tanto la velocidad de convergencia como el rendimiento final de la red. Una de las estrategias de tasa de aprendizaje más simples es tener una tasa de aprendizaje fija durante todo el proceso de formación. La elección de una tasa de aprendizaje pequeña permite al optimizador encontrar buenas soluciones, pero esto se produce a expensas de limitar la velocidad inicial de convergencia. Cambiar la tasa de aprendizaje con el tiempo puede superar esta compensación, ayudando a encontrar convergencia en redes complejas de entrenar como las redes generativas. Por tanto se experimentó con una función variable a lo largo del tiempo con forma triangular.

2.2. Preprocesamiento de Datos

Uno de los grandes beneficios que poseen los modelos de inteligencia artificial, es la capacidad de procesar una gran cantidad de datos y así encontrar correlaciones complejas difíciles de identificar para un humano. Sin embargo es necesario realizar una correcta elección de la información que se le entrega al modelo. El comportamiento de una empresa está influido por un gran número de factores externos, los cuales están interrelacionados.

A continuación se presentan aquellos factores considerados en esta investigación..

- a) Activos Relacionados : Estos son otros activos de cualquier tipo tales como acciones , divisas, índices globales, etc.
- b) Indicadores Técnicos : Como la bolsa de valores es una interacción humana de compra y venta, el análisis de indicadores forma parte importante de la toma de decisiones de muchos inversores.
- c) Autoregressive Integrated Moving Average (ARIMA) : Es un modelo estadístico que utiliza variaciones y regresiones de datos estadísticos con el fin de encontrar patrones para una predicción hacia el futuro. Se trata de un modelo dinámico

de series temporales, es decir, las estimaciones futuras vienen explicadas por los datos del pasado y no por variables independientes. Este modelo fue una de las técnicas más populares, en épocas pre neuronales, presentando una perspectiva estadística robusta y eficiente para series financieras especialmente en predicciones de corto plazo (Ariyo, Adewumi & Ayo, 2014). Sin embargo estudios demuestran que herramientas de deep learning como las redes LSTM entregan mejoras en un 85 % en promedio con respecto a los resultados provenientes de ARIMA (Siami-Namini & Namin, 2018). Es por ello que en esta investigación se combinan ambas técnicas entregando a los modelos de redes neuronales en estudio esta información como una variable de entrada.

- d) Componentes de Fourier : La transformadas de Fourier sirve para generalizar varias tendencias a largo y corto plazo. Usando estas transformaciones es posible eliminar ruido y crear aproximaciones del movimiento real del activo. Tener aproximaciones de tendencias puede ayudar a la red LSTM a elegir sus tendencias de predicción con mayor precisión.
- e) Análisis de Sentimiento en Fuentes Financieras: Las noticias pueden indicar eventos próximos que potencialmente pueden mover las acciones en cierta dirección. Dado que muchos inversores leen atentamente las noticias y toman decisiones de inversión basadas en las noticias, se considera una variable fuerte a considerar.

Considerando estos cinco aspectos contextuales junto con el precio diario de cierre, apertura, máximo , mínimo y volumen se obtiene una amplia cantidad de variables de entrada para los modelos propuestos en esta investigación. Para facilitarle la tarea a nuestros modelos, es necesario seleccionar aquellas variables que presentan mayor relevancia para el precio de cierre de cada activo. Para esta clasificación, se propone implementar la librería/algoritmo eXtreme Gradient Boost (XGBOOST). Este es uno de los algoritmos supervisados de aprendizaje automático que más se usan en la actualidad, debido a su fácil implementación junto con los buenos resultados en poco tiempo de procesamiento. Si el lector desea profundizar, diríjase a (Bentéjac, Csörgő & Martínez-Muñoz, 2019).

2.3. Aprendizaje Reforzado

Es un área del aprendizaje automático inspirada en la psicología conductual, donde se busca determinar qué acciones debe escoger un agente de software en un entorno

dado con el fin de maximizar alguna noción de “recompensa” o premio acumulado. Para esto no hay una indicación exacta de cómo se debe comportar el agente, más bien este evalúa los posibles estados futuros y aprende de los beneficios que se obtienen al realizar una acción, conceptualizando así el auto-aprendizaje. En este tipo de aprendizaje, existen 4 elementos fundamentales:

- a) Agente : El programa que se entrena, con el objetivo de hacer la tarea especificada.
- b) Entorno : El mundo, real o virtual, en el que el agente realiza acciones.
- c) Acción (A): Un movimiento realizado por el agente, que provoca un cambio de estado en el entorno.
- d) Recompensa (R): La evaluación de una acción dentro de un entorno, esta puede ser positiva o negativa.

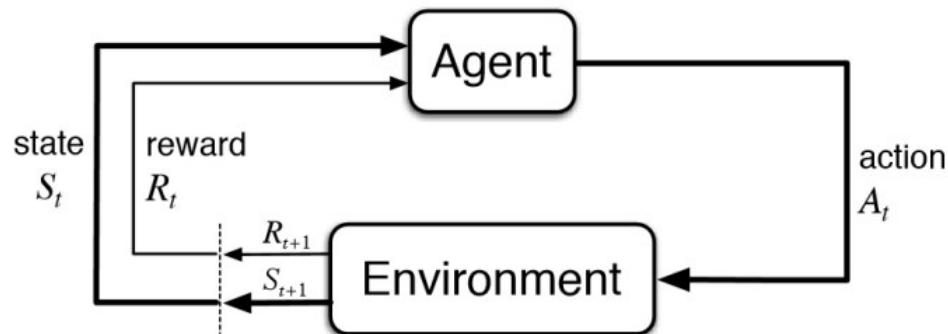


Ilustración 2.7: Proceso de Aprendizaje Reforzado

La combinación de este tipo de aprendizaje con las **redes neuronales profundas** se llama aprendizaje reforzado profundo, donde la acción siguiente está determinada por una red neuronal profunda cuyos datos de entrada son los estados actuales junto con la recompensa entregada por el entorno.

Mediante aprendizaje reforzado profundo es posible construir un entorno que entrene y evalúe la arquitectura propuesta por un agente en donde la recompensa sea una métrica de monitoreo que se pretende optimizar (WU, CHEN & CHEN, 2019) . De esta manera el agente determina los valores de ciertos parámetros dentro de un rango (tanto los parámetros como los rangos son establecidos por el programador) y el entorno devuelve el valor de recompensa obtenida a partir de esa elección. Mediante este paradigma se realizó la optimización de hiperparametros de las redes neuronales predictivas en este trabajo.

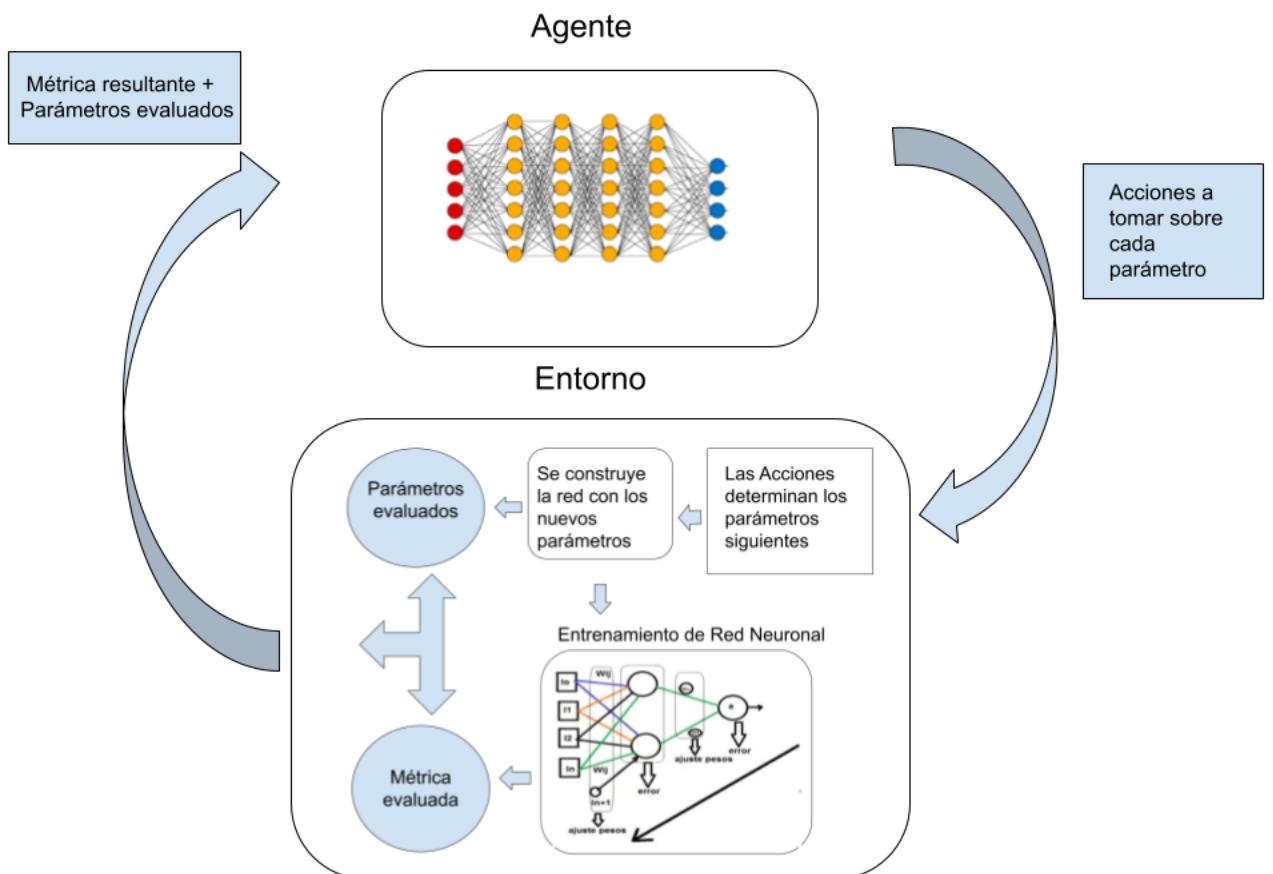


Ilustración 2.8: Implementación de Aprendizaje Reforzado Profundo para Optimización de Hiperparametros

Por otro lado, este tipo de redes son altamente utilizadas en la toma de decisiones financieras. Para esto se construye un entorno que evalúe las acciones del agente a partir de las ganancias obtenidas en un proceso de compra y venta de activos. Como el objetivo es maximizar las ganancias, la recompensa se le atribuye al excedente generado a partir de dichas acciones (Shah, Tambe, Bhatt & Rote, 2020). Esta simulación fue implementada en esta investigación con la finalidad de evaluar los modelos predictivos en estudio.

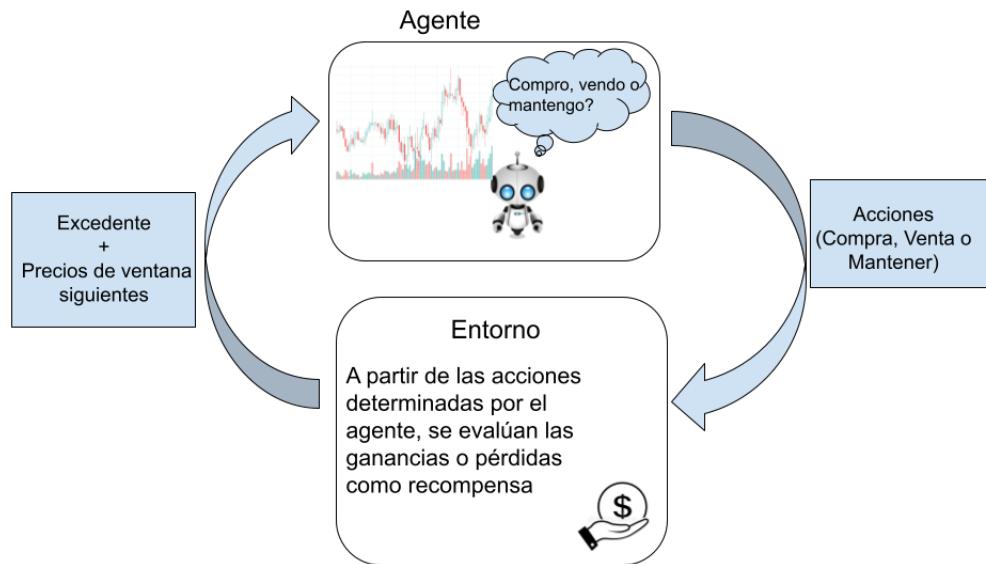


Ilustración 2.9: Implementación de Aprendizaje Reforzado Profundo para entorno financiero

2.4. Clasificación de Sentimiento en Noticias Financieras

Un clasificador de sentimiento es un modelo de aprendizaje supervisado que permite clasificar un texto **es** sensaciones positiva, negativa o neutra. Para esta tarea en particular, es posible obtener muy buenos resultados gracias a la transferencia de aprendizaje.

En particular en el área financiera existe **FinBERT** (Araci, 2019). Este modelo es una arquitectura basada en BERT⁹ (Devlin, Chang, Lee & Toutanova, 2019), cuyo objetivo específicamente es el análisis de sentimiento en noticias financieras. Para ajustar un modelo de lenguaje basado en BERT que cumpla la tarea de clasificación se le agrega una capa de clasificación después del token¹⁰ especial [CLS] , que se usa para tareas secuenciales como clasificación de oraciones o vinculación textual. Luego, todo el modelo se ajusta con pérdidas de clasificación. Una representación visual de esta estructura se encuentra en la siguiente imagen.

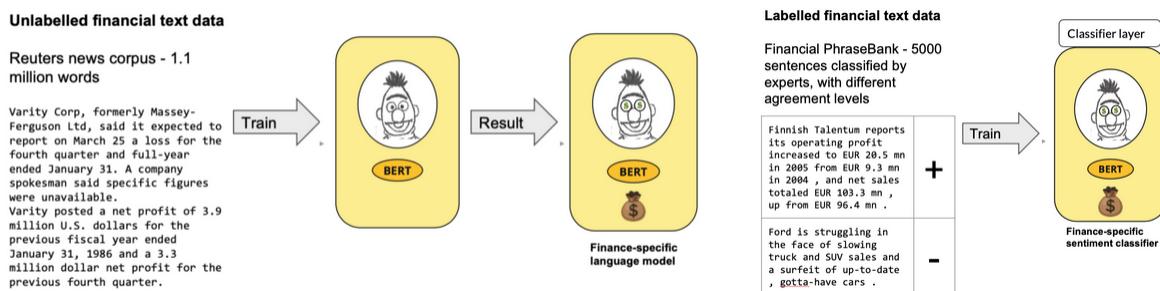


Ilustración 2.10: La descripción general de los pasos para capacitar a FinBERT

De esta manera se entrena el modelo original BERT con el conjunto de datos Reuters news corpus (TRC2) para que aprenda el dialecto financiero y luego con la adición de la capa de clasificación se entrenó con el conjunto de oraciones Financial PhraseBank (Malo, Sinha, Takala, Korhonen & Wallenius, 2013).

A pesar de que sus resultados son **sorprendentes** (97% en el subconjunto de acuerdo absoluto de Financial PhraseBank), los autores lograron identificar ciertas fallas **recurrentes**.

⁹Su arquitectura se basa en Transformadores. Si el lector desea profundizar mas en Transformadores dirijase al Anexo C.

¹⁰Son las piezas pequeñas que genera el proceso de Tokenización (Glosario)

- El modelo a veces falla en hacer las matemáticas en las que la cifra es más alta y, en ausencia de palabras indicativas de dirección como "aumentado", podría hacer una predicción neutral.
- El modelo no distingue una declaración neutral sobre una situación dada de una declaración que indica polaridad sobre la empresa.

El 73 % de las clasificaciones erróneas de FinBERT se encuentran entre etiquetas positivas y neutrales, mientras que solo el 5 % es para negativas y positivas. Eso es consistente tanto con los números de acuerdo entre los anotadores como con el sentido común. Es más fácil diferenciar entre positivo y negativo, pero podría ser más difícil decidir si una declaración indica una perspectiva positiva o simplemente una observación objetiva de carácter neutral.

2.4.1. Entrenamiento adversario para un clasificador de sentimientos

Con el fin de entregar mayor robustez al modelo pre-entrenado FinBERT sin la necesidad de crear un nuevo conjunto de datos de frases clasificadas, se propone realizar un entrenamiento léxico adversario mediante aprendizaje reforzado ("Xu, Zhao, Yan, Zeng, Liang & Sun, 2019). Como en todo entrenamiento adversario, el generador aprende a fabricar muestras para engañar al discriminador que en este caso es el mismo clasificador de sentimiento. Estas muestras son generadas a base de conocimiento léxico a gran escala (Word-net) reemplazando algunas palabras de ejemplo en el conjunto de entrenamiento con sinónimos, palabras vecinas o palabras superiores solo para cambiar la intensidad de la muestra y no el sentimiento.

Debido a que el generador crea muestras en forma discreta, es decir determina qué acción realizar para cada palabra de la oración, no es posible utilizar el enfoque del gradiente en forma directa para propagar el error. Por lo tanto se utiliza el gradiente de política como retroalimentación , donde la recompensa está condicionada a la función de coste en el clasificador . De esta manera el generador mejora en hacer las variaciones en cada oración obligando al clasificador a encontrar información fundamental para discernir.

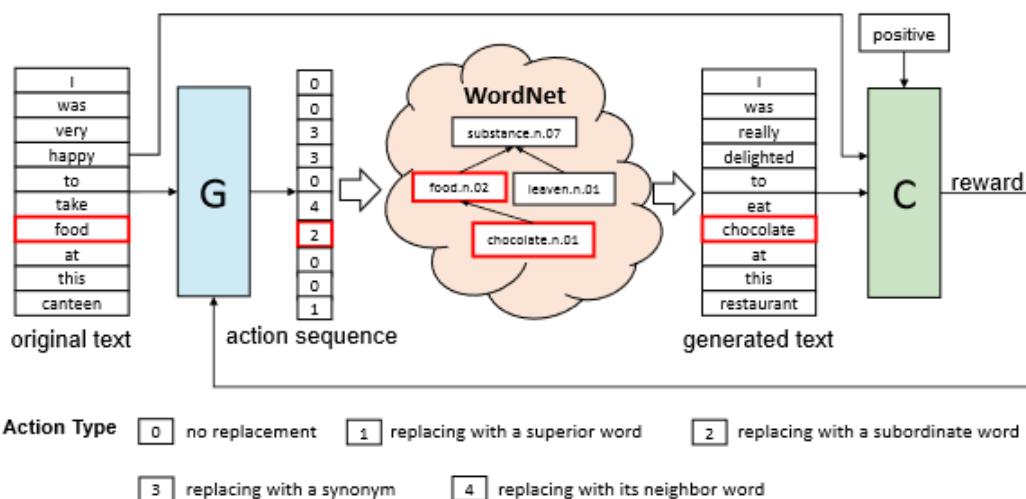


Ilustración 2.11: Entrenamiento Léxico Adversario, ("Xu, Zhao, Yan, Zeng, Liang & Sun, 2019)

Capítulo 3

Metodología

En este capítulo, se presenta una descripción de las diferentes etapas de la metodología propuesta para procesar información y así obtener predicciones en el comportamiento de activos con utilidad en decisiones financieras. En la Ilustración 3.1 se muestra el esquema metodológico como un diagrama de bloques.

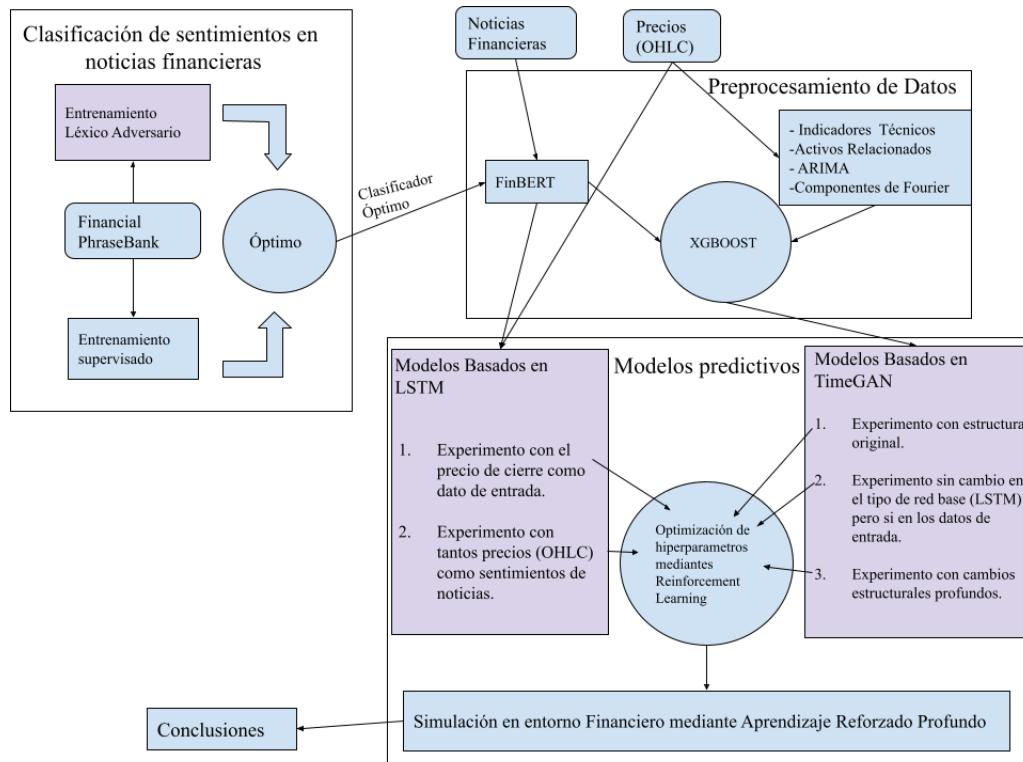


Ilustración 3.1: Diagrama de bloque de las diferentes etapas de la Metodología

Como se indica en la sección 1.2 , el objetivo principal de este trabajo es usar diferentes herramientas de vanguardia en inteligencia artificial y procesamiento de datos

en distintos escenarios experimentales con el propósito de evidenciar información útil y relevante en la toma de decisiones en determinadas actividades financieras. La idea principal para lograr este objetivo es dividido en tres áreas donde se cubren los 5 objetivos específicos descritos en la sección 1.2.1 :

- a) Clasificación de sentimiento en noticias financieras : El objetivo fundamental en esta etapa es implementar un clasificador de sentimiento financiero, con un nivel de **robustez aceptable**. Por tanto se realizaron dos tipos de entrenamiento al modelo FinBERT mencionado en la sección 2.4 con el conjunto de datos Financial PhraseBank. En primera instancia se realiza el entrenamiento supervisado descrito en la Ilustración 2.10. Luego se realiza un entrenamiento léxico adversario descrito en la sección 2.4.1 cuyo objetivo principal es aumentar la robustez en la clasificación del modelo entrenado inicialmente. Es por ello que este entrenamiento se realiza tanto para el modelo **pre-entrenado** con el conjunto TRC2 como el ya entrenado en forma supervisada . Finalmente se comparan los modelos con mejores resultados en el conjunto de prueba para así utilizar el clasificador de sentimientos óptimo en las noticias recopiladas. Luego de numerosos entrenamientos con diferentes parámetros no se logró garantizar una mejora en los modelos entrenados en forma léxico adversaria.
- b) Preprocesamiento de Datos : En esta etapa, el objetivo principal es procesar los datos recopilados en forma eficiente para los modelos predictivos. Es por ello que a partir de los datos en bruto se obtiene la información contextual descrita en la sección 2.2. A partir de los datos de precios se calculan los indicadores técnicos, ARIMA , componentes de fourier y activos relacionados mientras que las noticias son evaluadas por el clasificador de sentimientos óptimo proveniente de la etapa anterior para extraer un valor numérico que los modelos predictivos puedan interpretar. A partir de este conjunto total de datos se busca identificar los factores que afectan en el precio de cierre. Con la Librería/Algoritmo XGBOOST es posible obtener valores de importancia según activo.
- c) Modelos predictivos : Esta etapa final tiene como objetivo el estudio de dos modelos utilizados en predicciones futuras y corroborar su utilidad en la toma de decisiones financieras. Por un lado se realizan estudios de arquitecturas basadas en redes LSTM, a partir de dos experimentos que se diferencian en las variables evaluadas . Debido a las limitaciones que presenta este tipo de modelos utilizando solo aprendizaje supervisado el aumento de información solo genera ruido, es por ello que en el primer experimento se evalúa solo el precio

de cierre y en el segundo experimento se evalúan tanto los precios (OHLC¹) junto con los sentimientos noticiosos. Por otro lado se estudia gradualmente las arquitecturas basadas en TimeGAN a partir de tres experimentos. El primero fue realizado con la arquitectura original descrita en la Ilustración 2.6 cuyo objetivo es situarnos en un punto de partida y así conocer el funcionamiento de este tipo de arquitectura. El segundo experimento tiene como objetivo ver el comportamiento del generador al evaluar datos reales de la ventana de tiempo previa. En el último experimento se busca realizar cambios a nivel estructural y así comparar resultados con el experimento dos. Para los cinco experimentos se realiza una búsqueda de hiperparámetros a partir de aprendizaje reforzado profundo descrito en la Ilustración 2.8. Si bien todos los experimentos anteriores se evalúan a partir de una métrica del porcentaje de acierto (Accuracy²), esta no garantiza que las predicciones tengan una utilidad real. Es por ello que se diseña una simulación en entorno financiero mediante aprendizaje reforzado profundo , descrito en la ilustración 2.9, para comparar la utilidad en las predicciones de los cinco modelos con mejor resultado cualitativo.



¹Del inglés Open, High, Low y Close (OHLC), representan los precios de apertura, máximo , mínimo y cierre.

²Es el porcentaje total de elementos clasificados correctamente, para nuestra investigación se realiza una diferencia porcentual de la ventana de tiempo evaluada

Capítulo 4

Clasificación de Sentimientos Financiero

En este capítulo, se presenta una descripción de las diferentes etapas realizadas para la implementación de un clasificador de sentimientos financieros, con un nivel de **robustez aceptable**. En la Ilustración 4.1 se muestra el esquema metodológico de este capítulo como un diagrama de bloques.

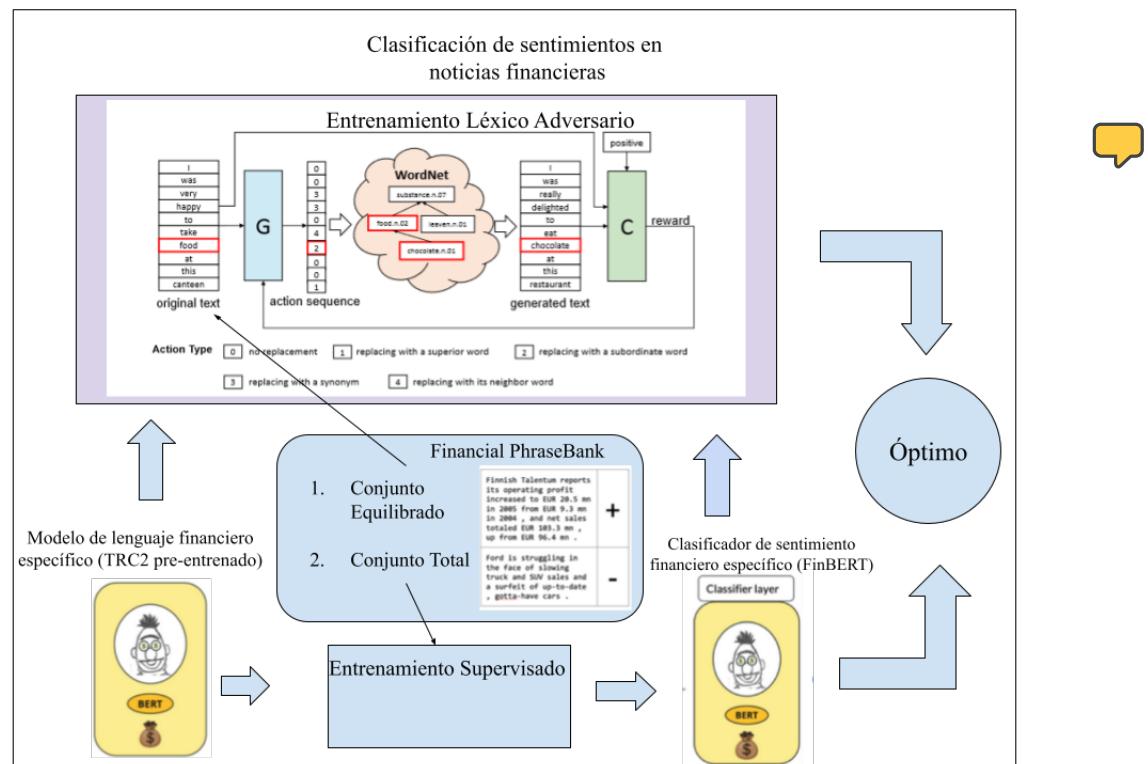


Ilustración 4.1: Esquema metodológico en la implementación del clasificador de sentimiento financiero

4.1. Datos

Se utilizó el conjunto de datos de Financial PhraseBank (Malo, Sinha, Takala, Korhonen & Wallenius, 2013). Este presenta una recopilación de frases características provenientes de fuentes financieras clasificadas por 16 expertos en finanzas. Tres de los anotadores eran investigadores, y los 13 anotadores restantes eran estudiantes de maestría en la Escuela de Negocios de la Universidad de Aalto con especializaciones principalmente en finanzas, contabilidad y economía. Luego de la evaluación de los expertos el conjunto total presenta cuatro subconjuntos de datos según la fuerza de acuerdo mayoritario, descritos en la tabla 4.1.

Subconjunto	Negativo %	Neutral %	Positivo %	Nº oraciones
con 100 % de acuerdo	13.4	61.4	25.2	2259
con más de 75 % de acuerdo	12.2	62.1	25.7	3448
con más de 66 % de acuerdo	12.2	60.1	27.7	4211
con más de 50 % de acuerdo	12.5	59.4	28.2	4840

Tabla 4.1: Distribución de etiquetas en banco de frases para 4 subconjuntos formados en base a la fuerza del acuerdo mayoritario. Cada oración tiene 5 a 8 anotaciones superpuestas, que se han utilizado para determinar el grado de acuerdo.

Lamentablemente, como se puede ver en la tabla 4.1, todos los subconjuntos presentan cantidades desproporcionadas de sentencias según tipo de sentimiento, donde el número de sentencias neutras es mucho mayor.

En particular para aquellas clasificadas negativamente se cuentan con un número muy bajo (605 oraciones en el conjunto de mayor número de oraciones ($\geq 50\%$) de concordancia), por eso se crea un conjunto de datos equilibrados de la siguiente manera.

- a) Para el conjunto de entrenamiento se utilizan 605 oraciones por tipo de sentimiento. Aquellas negativas a partir del conjunto con más del 50 % de concordancia, las positivas provienen del conjunto con más del 75 % de concordancia y las neutras del conjunto con total concordancia.
- b) Para el conjunto de testeo se utiliza 200 oraciones de cada sentimiento provenientes del conjunto con 100 % de concordancia. Se busca no repetir oraciones presentes en el conjunto de entrenamiento pero para el caso de las oraciones negativas no es posible (todas fueron utilizadas en el conjunto de entrenamiento).

- c) Para el conjunto de evaluación solo se utilizan oraciones positivas y neutras (70 de cada tipo también proveniente del conjunto con total concordancia pero no evaluadas anteriormente).

Solo se utilizan oraciones positivas y neutras en el conjunto de evaluación por dos motivos. Por un lado, se desea evaluar los modelos entrenados en forma léxico adversaria solo con oraciones nuevas por lo que no existen oraciones con sentimiento negativo que no hayan sido evaluadas anteriormente. Por otro lado, la mayor dificultad para el modelo FinBERT original está en discernir entre estos dos sentimientos. Por tanto, como nuestro objetivo es mejorar el rendimiento del modelo FinBERT original, se busca demostrar un mejor rendimiento en aquellas oraciones de mayor dificultad .

4.2. Entrenamiento Supervisado

El entrenamiento supervisado se realizó a partir de los pesos pre-entrenados con el conjunto de datos TRC2 con la totalidad del conjunto de datos con acuerdo superior al 50 % ¹.

4.3. Entrenamiento Léxico Adversario

Para el entrenamiento léxico adversario, se utilizan 2 puntos de partida.

- a) A partir del clasificador ya entrenado en forma supervisada descrita en la sección anterior.
- b) A partir del corpus TRC2, proporcionado por FinBERT. El mismo que se utilizó para el entrenamiento supervisado.

Si bien el primer punto de partida ya evalúa el conjunto de oraciones originales en el entrenamiento supervisado lo que podría provocar un fenómeno llamado overfitting ². Se asume que con el entrenamiento léxico adversario, el cual busca enseñar al generador a hacer cambios de palabras dentro de la oración original, el modelo no evaluará el mismo conjunto de datos. Si el supuesto fuese erróneo, es decir el clasificador presente sobreajuste, el entrenamiento léxico adversario con el segundo punto de partida (TRC2 pre-entrenado) permite extraer conclusiones comparativas con un

¹De esta manera se recomienda por los autores de FinBERT (Araci, 2019).

²El efecto de sobreentrenar un algoritmo de aprendizaje con ciertos datos para los que se conoce el resultado deseado.

modelo que nunca evaluó las oraciones originales.

Para los cambios de sentencias, como las variaciones pueden distorsionar demasiado el sentido de la oración, se utilizaron 3 tipos de posibles cambios que aumentan en forma gradual las capacidades del generador en alterar la oración :

- a) Tipo 1: Solo cambios de sinónimos.
- b) Tipo 2: Cambios de sinónimo, hipónimos e hiperónimos.
- c) Tipo 3: Todos los cambios posibles, es decir sinónimos, hipónimos, hiperónimos y ambos en forma secuencial..

Con respecto al conjunto de datos, en primera instancia se realizan pruebas con el mismo conjunto de datos utilizados para el entrenamiento supervisado, cuyos resultados presentan notorias tendencias a categorizar en forma neutral todas las sentencias. Por tanto se utiliza el conjunto de datos equilibrados para todos los entrenamientos léxico adversarios posteriores.

Con respecto a la metodología de entrenamiento, se aumenta en forma gradual tanto el número de épocas como el tipo de cambios permitidos al generador.

A continuación se presentan los resultados de algunos entrenamientos léxico adversario, dentro de los cuales se encuentran los modelos de mejor rendimiento en la etapa de testeo.

Modelo pre-entrenado	épocas	cambio	Accuracy Test
TRC2	9	1	0.2418867
TRC2	9	2	0.3418867
TRC2	9	3	0.4018867
TRC2	30	3	0.8566667
TRC2	40	3	0.88653334
TRC2	50	3	0.9066667
FinBERT	9	1	0.24133658
FinBERT	9	2	0.30166667
FinBERT	9	3	0.44166667
FinBERT	30	3	0.86833334
FinBERT	40	3	0.86333334
FinBERT	50	3	0.885

Tabla 4.2: Resultados de Entrenamientos Léxico Adversarios

En la tabla 4.2 se logra ver, que con solo nueve épocas, en ambos puntos de partida se presenta un mayor nivel de acierto al permitir todo tipo de cambios. Es

por ello que los entrenamientos siguientes se realizan con máxima libertad de cambio para el generador. También es posible ver notorias mejoras a medida que aumenta el número de épocas. Sin embargo a partir de las treinta épocas, dichas mejoras se vuelven pequeñas encontrando una asíntota en entrenamientos mayores o iguales a cincuenta épocas. Todos los entrenamientos con más de cincuenta épocas no entregan resultados mejores a los expuestos en la tabla 4.2.

Si bien en ambos puntos de partida los entrenamientos de mejor resultado son los de cincuenta épocas, la mejora no es sustancial en relación a los de treinta y cuarenta épocas. Es necesario considerar que estos resultados fueron a partir de un conjunto de testeo que presenta oraciones utilizadas en el conjunto de entrenamiento. Esto nos lleva a pensar que la pequeña mejora se obtiene debido a la similitud de la oración vista en más oportunidades, lo que no me refleja necesariamente a una mayor capacidad de discernir en sentimiento.

4.4. Análisis de Evaluación Comparativo

Se realiza una comparación con oraciones positivas y neutras nunca antes visto en el entrenamiento léxico adversario. Esta evaluación considera el modelo finBERT original junto con los tres mejores modelos entrenado léxico adversaria mente de ambos puntos de partida, es decir tanto para el punto de partida FinBERT supervisado como a partir de TRC2 se consideran los entrenamientos de treinta, cuarenta y cincuenta épocas. Los resultados de la evaluación comparativa se describen en la siguiente tabla, donde aquellos entrenados léxico adversaria mente se engloban por punto de partida

Modelo pre-entrenado	Accuracy Evaluación
FinBERT original	0.6255924170616114
FinBERT adv	0.5876777251184834
TRC2 adv	0.5781990521327014

Tabla 4.3: Resultados de entrenamientos para clasificador de sentimiento de noticias financieras

Si bien los resultados presentados en la tabla 4.2 le atribuyen mejores clasificaciones al modelo construido a partir del corpus TRC2 logrando hasta 90 % de acierto en el conjunto de testeo en relación al 88 % del entrenado a partir del clasificador FinBERT, esto no se mantiene en el conjunto de pruebas comparativas donde el segundo logra un resultado levemente superior. Es importante señalar que el conjunto de evaluación

presenta una limitada cantidad de oraciones (solo 140 oraciones), por lo que esta pequeña diferencia no implica una falta de correlación entre ambos resultados. Esta pequeña mejora se puede atribuir al entrenamiento supervisado realizado previo al entrenamiento léxico adversario, donde este punto de partida evalúa el total de oraciones presentes en el conjunto de evaluación por lo que sus pesos fueron ajustados inicialmente considerando estas oraciones.

Como se explica en la sección 2.4, diferenciar entre oraciones positivas y neutras es de alta complejidad, lo cual justifica los bajos niveles de acierto global, reflejado en la tabla 4.3.

Ningún modelo entrenado en forma léxico adversaria supera los resultados obtenidos por el modelo FinBERT original. Esto se puede atribuir a la imparcialidad en la evaluación comparativa, ya que los datos evaluados nunca fueron vistos por los entrenamientos léxico adversarios pero si fueron evaluados en el entrenamiento supervisado. De esta manera el modelo que presenta peor resultado es aquel que nunca vio ninguna oración del conjunto de evaluación .

La limitada cantidad de datos condiciona la evaluación comparativa. Por consiguiente, para poder realizar evaluaciones válidas, es necesario un conjunto diferente de oraciones financieras que ningún modelo haya tenido la posibilidad de aprender. Para esto se requiere de la creación de un nuevo conjunto de datos clasificados por expertos financieros, lo cual no fue posible conseguir en este trabajo.

De todos modos, debido a la falta de evidencia en el aumento de robustez, se decidió utilizar el modelo FinBERT original para determinar el sentimiento de las noticias recopiladas.

Capítulo 5

Preprocesamiento de Datos

En el presente capítulo se describen los procesos de recopilación y preprocesado de los Datos utilizados en los modelos predictivos. En primer lugar se describe tanto la recopilación como el tipo de dato que se utilizó. En segundo lugar se describe la categorización de importancia en las variables de estudio y finalmente las características de los datos utilizados en ambos modelos predictivos, tanto en los modelos basados en LSTM como en los basados en redes generativas adversarias.

5.1. Recopilación de Datos

Para juntar una cantidad considerable de datos, se creó un sistema autónomo de extracción y almacenado de precios y noticias financieras por activo. A partir de una base de datos relacional se asocia dos tablas hijas a partir de una tabla ancla con los nombres de los activos pertenecientes al DOW JONES junto con una variable de identificación. Una de estas tablas hijas almacena solo variables de precio diarios (OHLC) recopilados a partir de la API de Yahoo Finance y la otra almacena las noticias financieras provenientes de 3 fuentes estadounidenses, Nasdaq.com, finviz.com y finance.yahoo.com.

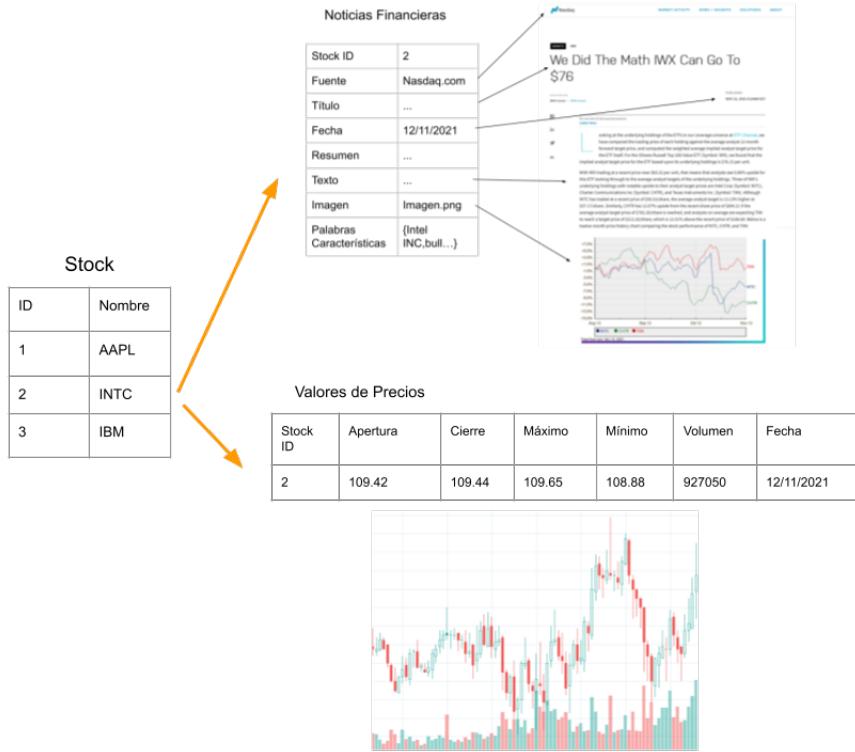


Ilustración 5.1: Estructura relacional de la base de datos

A partir de esta información fundamental, se obtienen todas las variables contextuales descritas en la sección 2.2. A continuación se describen los detalles de cada uno de los aspectos contextuales.

5.1.1. Activos Relacionados

En primer lugar se integran índices globales relevantes. De esta manera se busca integrar información de conjunto de activos estadounidense (SP500, Nasdaq , Dow Jones, NYSE , Russell 2000), europeos (Cboe UK 100, FTSE, DAX 30, CAC 40, euronext 100, BEL20, MOEX Russia), japonés (Nikkei 225), chino (Hang Seng Index, SSE Composite ,The SZSE Component) entre otros.

En segundo lugar se desea identificar qué activos, dentro del mismo índice bursátil (Dow Jones), presentan mayor relación en su comportamiento. Finalmente, como en un sistema globalizado el valor de aquellos activos en estudio dependen del flujo global de recursos, se consideran 13 divisas de las economías globales más importantes junto con la criptomoneda de mayor importancia en la actualidad, el Bitcoin .

Nombre	Intercambio
EUR/USD	proporción entre el Euro y el dólar
AUD/USD	proporción entre el dólar australiano con el dólar
CHF/USD	proporción entre el franco suizo con el dólar
GBP/USD	proporción entre la Libra esterlina con el dólar
JPY/USD	proporción entre el Yen japonés con el dólar
NZD/USD	proporción entre el dolar neozelandes con el dólar estadounidense
BTC/USD	proporción entre el Bitcoin con el dólar
CNY	Yuan chino
HKD	Dolar HONKONES
RUB	Rublo Ruso
IDR	Rupia Indonesia
SGD	Dólar Singapourense
INR	Rupia India
MXN	Peso Mexicano

Tabla 5.1: Tipo de monedas Globales evaluadas en esta investigación

Es importante recalcar que toda esta información fue posible obtenerla gracias a la API de yahoo finance que proporciona una basta información de la economía mundial.

5.1.2. Indicadores Tecnicos

Este tipo de indicadores son herramientas fundamentales para aquellos participantes que toman sus decisiones basados en análisis técnico. Por ello se opto por considerar los más populares (MA7, MA21, Momentum, EMA, MACD, ATR, Bandas de Bollinger, RSI) ¹.

Nombre	Definición
MA7	Media Movil con ventana de 7 días
MA21	Media Móvil con ventana de 21 días
EMA	Media Móvil Exponencial
MACD	Media Móvil Convergencia/Divergencia
ATR	Promedio del Rango Verdadero
RSI	Indice de Fuerza Relativa

Tabla 5.2: Definición de indicadores técnicos

¹Las funciones de los indicadores tecnicos se pueden encontrar en : https://github.com/chomiestyle/Analisis_tecnico/blob/master/Work/TechnicalAnalysis.py

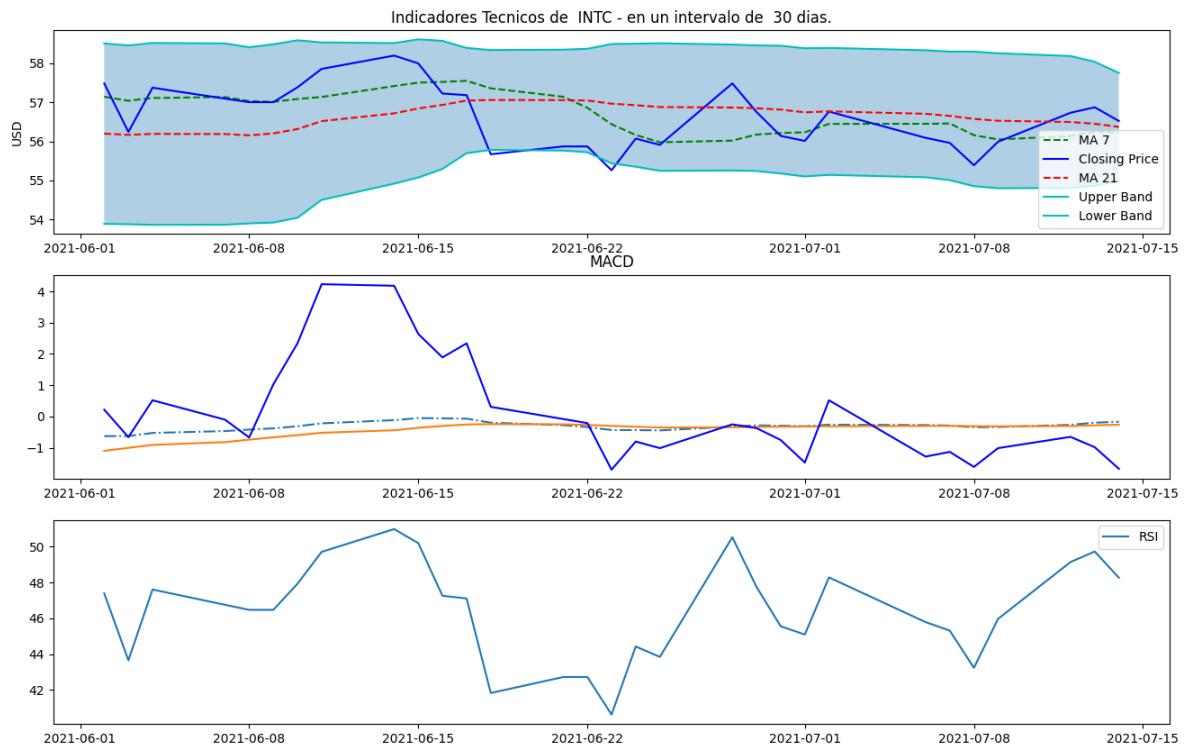


Ilustración 5.2: Gráficos de Indicadores Técnicos para el precio de cierre de 'INTC'

5.1.3. Modelo Autorregresivo Integrado de Media Móvil

Los resultados presentes en la Ilustración 5.3², donde se obtuvo un error cuadrático medio (MSE) de 0.891, permiten evidenciar los buenos resultados entregados por este modelo. Es por ello que en esta investigación, como fue señalado en la sección 2.2, se pretende entregar a los modelos de redes neuronales en estudio (específicamente aquellos basados en redes neuronales generativas adversarias) las predicciones del precio de cierre realizadas por ARIMA como una posible variable de entrada. De esta manera se busca que el generador utilice los patrones recaudados por un modelo estadístico para generar nuevos patrones y así engañar al discriminador.

²El entrenamiento del modelo fue realizado a partir del precio de cierre de 'INTC' desde 2010-01-01 hasta 2021-07-15, cuya evaluación se realiza desde 2021-07-15 hasta 2021-11-23.

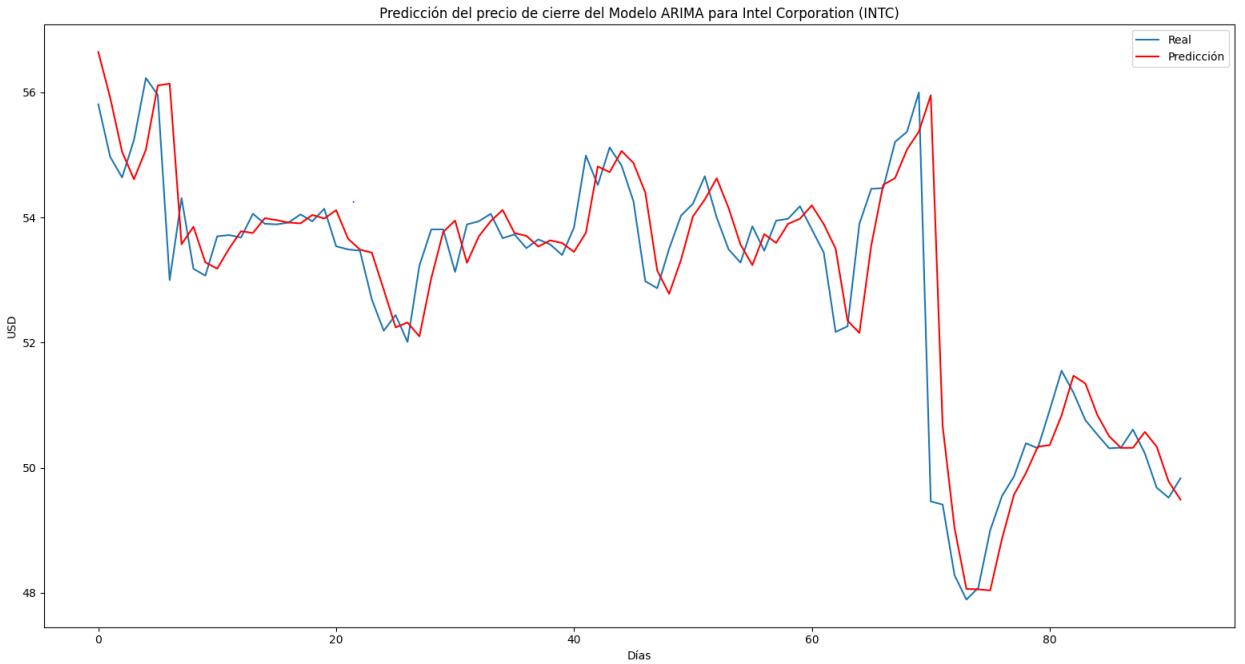


Ilustración 5.3: Predicción del precio de cierre del Modelo ARIMA para Intel Corporation (“ INTC ”)

5.1.4. Componentes de Fourier

Las transformadas de Fourier nos permiten interpretar la dinámica de una serie de tiempo en el rango de las frecuencias. Una serie de tiempo es , en el fondo, la suma de muchas ondas sinusoidales . Por tanto esta transformada toma una función o serie temporal y crea una serie de ondas sinusoidales (con diferentes amplitudes y frecuencias). Dicho de otra manera, la transformada de fourier es un separador de series temporales en ondas simples. Estas ondas simples permiten extraer información estructural de la señal.Es por eso que usamos transformadas de Fourier de 3, 6 y 9 componentes para extraer tendencias globales y locales en los activos.

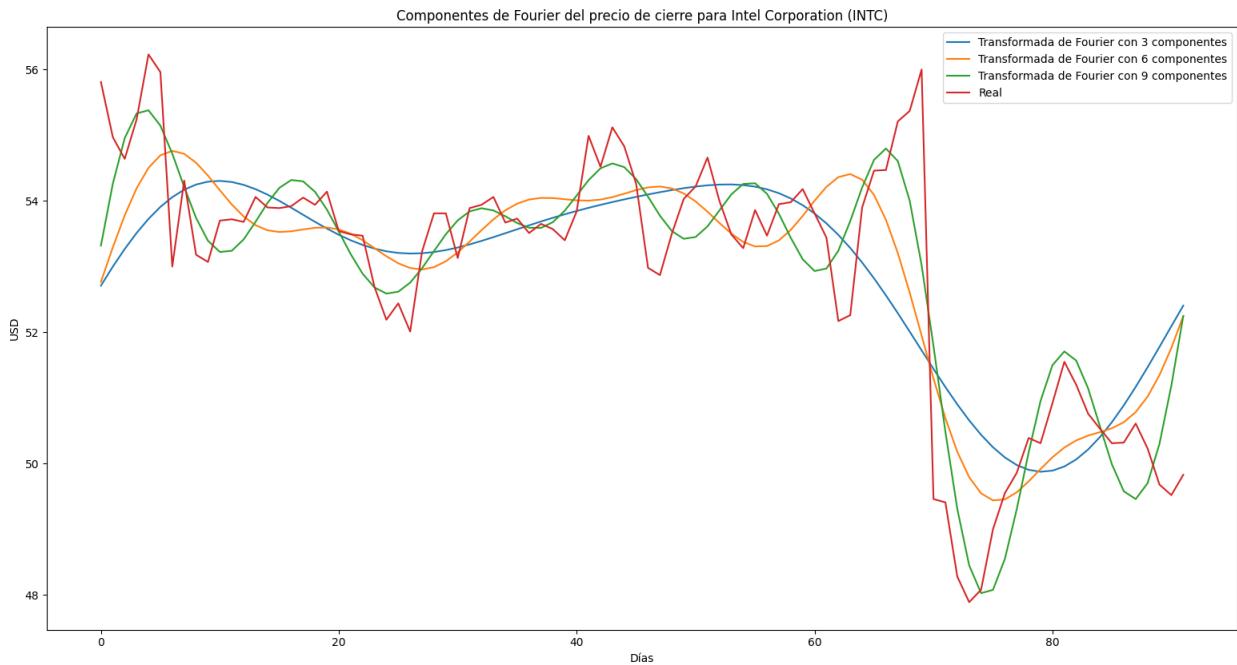


Ilustración 5.4: Componentes de Fourier del precio de cierre para Intel Corporation (“INTC”)

5.1.5. Análisis de Sentimiento en Fuentes Financieras

Una vez se tiene el clasificador de sentimientos definitivo, descrito en la sección 4.4, la categorización se realiza en la información almacenada en la tabla de noticias financieras (5.1). Debido a la gran cantidad de noticias y el tiempo que necesitan para ser clasificadas, los resultados fueron almacenados en una tabla hija de la tabla de noticias financieras como se describe en la Ilustración 5.5. A partir de los valores de esta tabla se obtiene el promedio y desviación estándar diario de los sentimientos de las tres estructuras base de la noticia (Título, Resumen y Cuerpo o Texto).

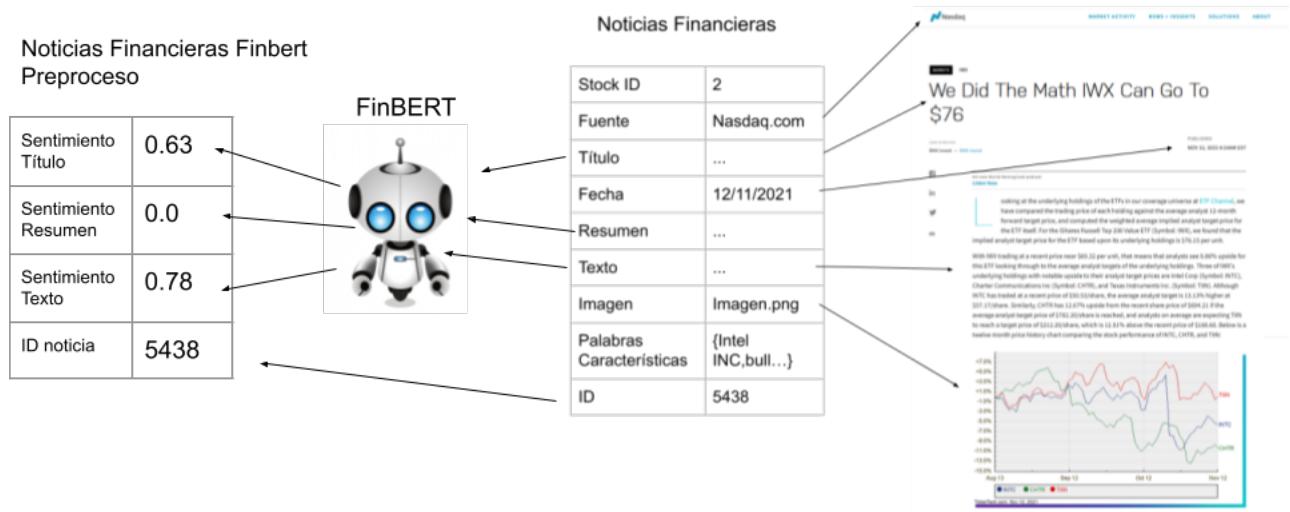


Ilustración 5.5: Diagrama del Análisis de Noticias Financieras.

5.2. Categorización de Variables según su Nivel de Importancia

Como presentamos una gran cantidad de variables, es necesario depurar la data según relevancia eliminando así aquellas variables que no presentan poder explicativo de la población que queremos predecir. Para este proceso se utilizó XGBoost (eXtreme Gradient Boosting), un tipo de algoritmos de regresión mediante árboles de decisión potenciados. Para cada proceso se realizó una búsqueda óptima de hyper parámetros según un set de entrenamiento ³.

³El algoritmos de optimización de hiperparametros se encuentra en : <https://github.com/chomiestyle/HPxgboost>

Los siguientes parámetros se ajustaron para XGBoost en este estudio:

- a) Tasa de aprendizaje (Learning rate) : La rapidez con que el modelo se adapta al problema.
- b) Reducción de pérdida mínima (Gamma) : Cuanto mayor sea este valor, menos profundos serán los árboles.
- c) Máxima Profundidad (Max depth) : La profundidad máxima del árbol.
- d) (Colsample bylevel): La fracción de características que se evaluarán en cada división.
- e) Tasa de submuestreo (Subsample) : El muestreo se realiza sin reemplazo.
- f) (n estimators) : El número de estimadores a utilizar.
- g) Menor peso de árboles hijos (Min child weight) : El peso mínimo que puede tener cada hoja hija.

Los valores óptimos, luego de la búsqueda de hiperparametros, se describen en la tabla 5.3.

Parámetro	Acciones	Indices Globales	Divisas	Indicadores Técnicos
Learning rate	0.1	0.1	0.1	0.065290
Gamma	0	5	0	1
Max depth	3	10	7	9
Colsample bytree	0.444094	0.3	0.7	0.866914
Subsample	0.5	0.517746	0.7	0.571136
n estimators	100	100	100	100
Min child weight	1	1	3	1

Tabla 5.3: Resultados de Optimización de Hiperparametros algoritmo XGBoost

En teoría la importancia depende del tipo de activo, por tanto es necesario realizar este estudio en forma cautelosa y dependiente del activo en estudio. La regresión se realiza con respecto al precio de cierre y en diferentes etapas.

En primer lugar, con respecto a los activos relacionados, se determinó los 10 factores más influyentes dentro de las empresas del DOW JONES. Para esto se evalúa por cada empresa, cualquier valor ya sea de cierre (Close), apertura (Open), máximo (High), mínimo (Low) y volumen (Volume) de todo el resto de los activos almacenados en la tabla de precios descrita en la ilustración 5.1 . De esta manera se pueden ver en la

ilustración 5.6 para el caso de “ INTC ” existe una alta relación con Disney (’DIS’) y Microsoft Inc (’MSFT’) sumado a valores de cierre de 3M Company (’MMM’), Johnson Johnson (’JNJ’) y McDonald’s (’MCD’).

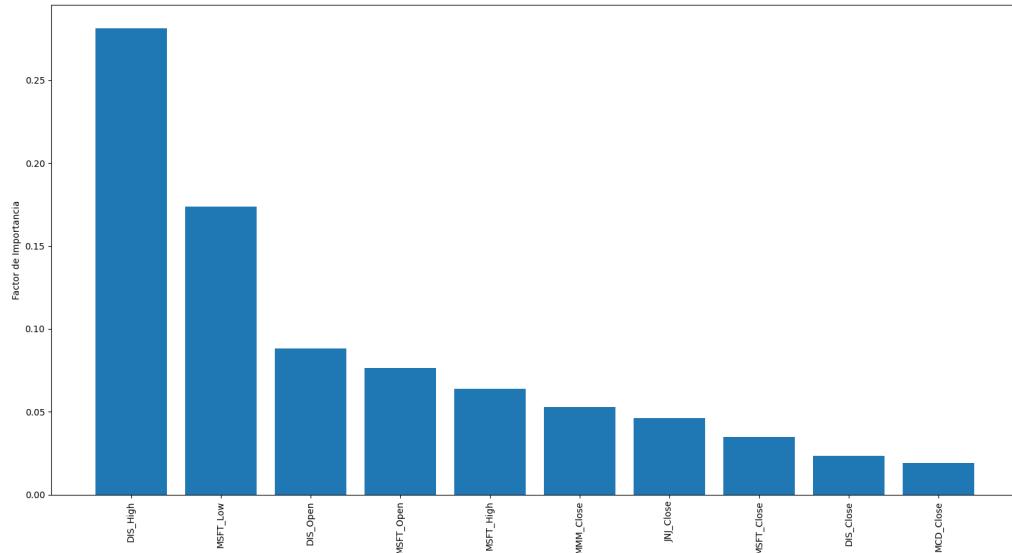


Ilustración 5.6: 10 Acciones del Dow Jones de mayor importancia para “ INTC ”.

También se realizó una búsqueda tanto en las divisas internacionales (descrita en la tabla 5.1) y los índices globales seleccionando los 10 factores más importantes de cada uno. Para el caso de “ INTC ” (Ilustración 5.7) los dos índices más importantes son el Bombay Stock Exchange Sensitive Index (BSE Sensex, acrónimo para el índice de la Bolsa de Valores de Bombay) y Nasdaq. Esto se puede atribuir a que ambos representan 2 economías globales muy importantes (India y EE.UU). Sin embargo llama la atención que MERVAL (el principal índice del Mercado de Valores de Buenos Aires) y SP/NZX 50 Index (principal índice bursátil de Nueva Zelanda) presenten mayor importancia que ciertos índices estadounidenses importantes como SP500 y el mismo DOW 30 . Con respecto a las monedas globales se puede ver en Ilustración 5.8 que monedas como el Rublo Ruso, Rupia Indones y la Libra Inglesa afecta en el precio de cierre de una empresa como Intel Inc.

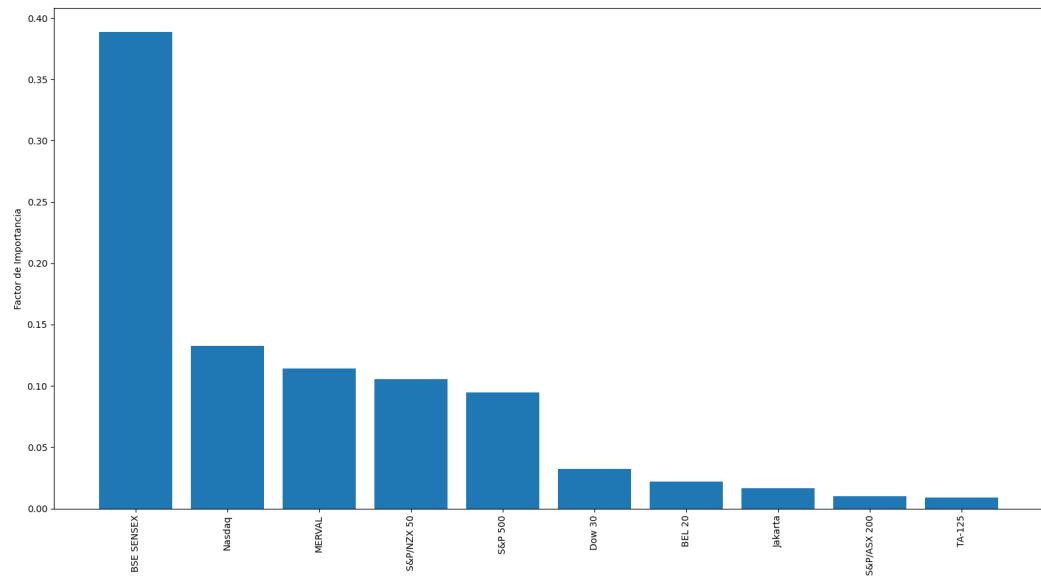


Ilustración 5.7: Los 10 Índices Globales de mayor Importancia para “ INTC ”.

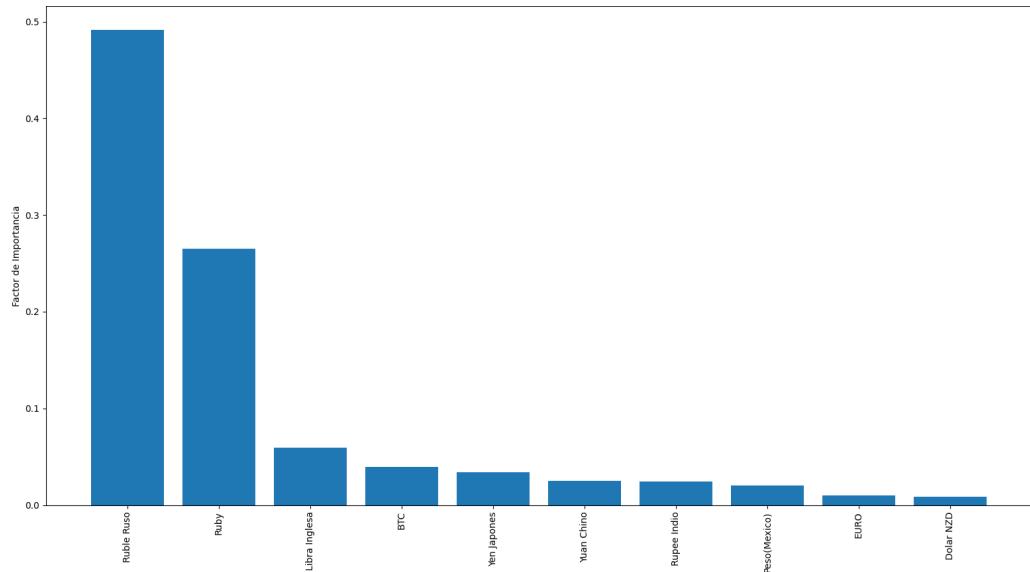


Ilustración 5.8: Las 10 Monedas Globales de mayor factor de relevancia para 'INTC'

En segundo lugar se evaluó la importancia de ciertos indicadores técnicos que podrían ser útiles. Por ejemplo para el caso particular de “ INTC ”, Ilustración 5.9 , se puede ver que los indicadores de medias móviles (MA7, MA21 y EMA) en su conjunto presentan un gran nivel de relevancia.

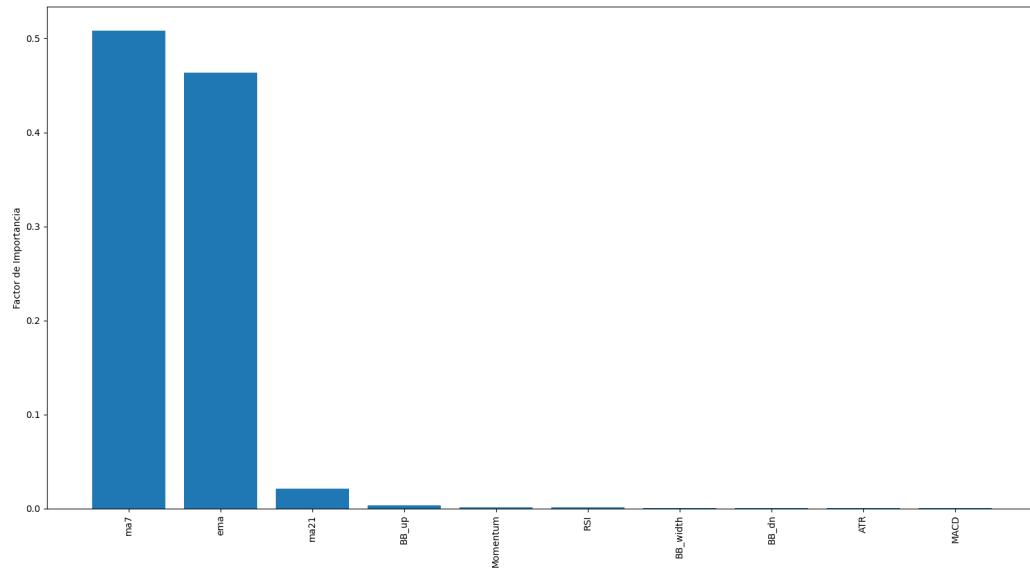


Ilustración 5.9: 10 Indicadores Técnicos con mayor relevancia para “ INTC ”

Finalmente se le agregaron los componentes de Fourier, ARIMA y factor de sentimiento noticioso diario creando así un conjunto de datos contextual. A partir de este conjunto de datos global se realiza un subconjunto más pequeño donde solo quedaron los 10 factores más influyentes a nivel global. Es importante recalcar que como la cantidad de noticias financieras reducen el tamaño total de datos, para los casos en donde no se refleja relevancia se utilizó los datos relevantes sin las limitaciones adquiridas por las noticias financieras.

Como se puede ver en la Ilustración 5.10 para el caso de Intel Incorporated (“ INTC ”), las variables de mayor importancia se encuentran dentro de las relevantes en cada subconjunto pasado pero no necesariamente reflejan el mismo orden. Esto se atribuye a que el algoritmo no analiza las variables en forma independiente sino a conjuntos de variables que pueden sumar más información relevante para el precio de cierre.

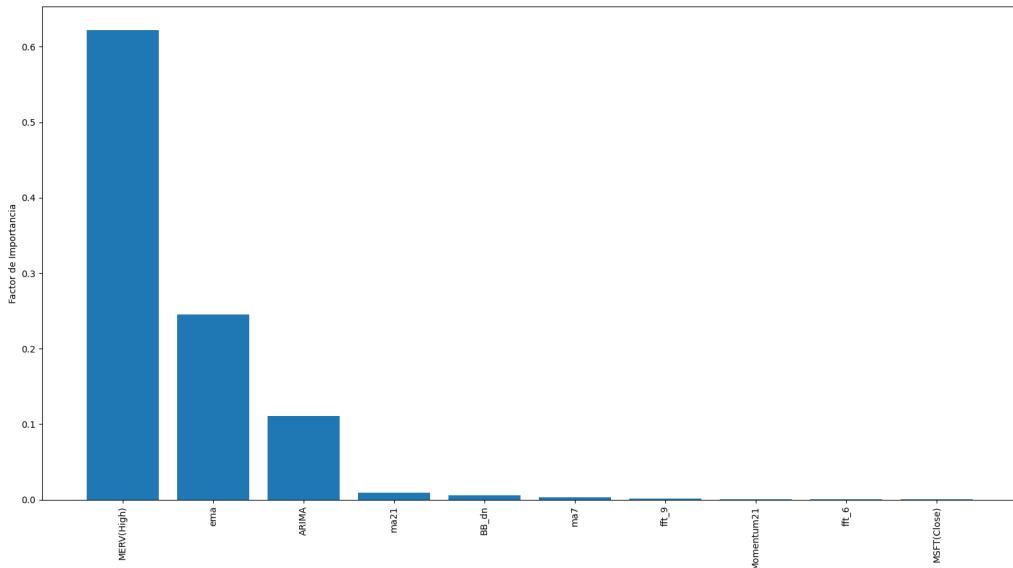


Ilustración 5.10: 10 variables contextuales con mayor relevancia para “ INTC ”

5.3. Características de Datos para Modelos Predictivos

Con respecto a los datos utilizados en los modelos predictivos basados en modelos LSTM, su aprendizaje supervisado tiene ciertas limitaciones en la capacidad de correlacionar información contextual. Por ello se optó por dos conjuntos de datos de entrada. Uno solo con el precio de cierre y otra con los valores de los precios (OHLC) junto con el promedio de sentimientos diarios.

Para el caso de los modelos de redes generativas adversarias, gracias a su entrenamiento semi-supervisado el aumento de variables de entradas le permite al generador evaluar un conjunto de datos con mayor aleatoriedad y difíciles de correlacionar logrando una propiedad creativa sustentada en datos reales. Es por ello que los datos de entrada en los experimentos asociados a este tipo de modelos presentan los datos preprocesados en la sección anterior.

Capítulo 6

Modelos Predictivos para Activos Financieros

En este capítulo, se describe en detalle el trabajo realizado en modelos predictivos financieros . Primero se exponen los experimentos realizados en arquitecturas de redes neuronales construidas únicamente con neuronas LSTM. En segundo lugar se presentan las tres etapas realizadas con redes neuronales generativas adversarias para series de tiempo junto con sus resultados cualitativos. Finalmente para comparar la utilidad de ambos modelos, se le entrega esta información futura a un agente de aprendizaje reforzado profundo para determinar las acciones en un entorno de compra y venta de activos.

6.1. Modelo con Arquitectura basada en Redes LSTM

Se busca obtener predicciones del precio de cierre, en ventanas semanales¹ a partir de modelos que presenten únicamente neuronas LSTM. Para esto se realiza una búsqueda óptima de hiperparametros a partir de entrenamiento Reforzado² . Se utiliza como métrica de monitoreo la diferencia porcentual entre el resultado predicho por el modelo con respecto a los valores reales en un lote .

¹Cuando hablamos de ventanas semanales, nos referimos a realizar predicciones de la semana próxima a partir de la actual semana

²Lo desarrollado en esta etapa fue basado en el trabajo presente en <https://github.com/lshalon/ezpeey>

```

def calculate_accuracy(y_true, y_pred):
    real = y_true + 1
    predict = y_pred + 1
    cuadrado=tf.square((real - predict) / real)
    axis = list(range(len(cuadrado.get_shape()) - 1))
    mean, var = tf.nn.moments(x=cuadrado, axes=axis)
    raiz=tf.sqrt(mean)
    percentage = 1 - raiz
    return percentage * 100

```

Ilustración 6.1: Funcion de Accuracy (Diferencia Porcentual) utilizado como métrica cuantitativa en todos los modelos predictivos.

Los parámetros a optimizar en esta etapa son:

Parámetro	rango
Número de capas LSTM	1-5
Número de neuronas en las capas LSTM	32-1024
Porcentaje de dropout en las capas LSTM	0.1-0.99
rango de aprendizaje	0.01-0.1

Tabla 6.1: Tabla de parámetros a optimizar en arquitecturas basadas en LSTM.

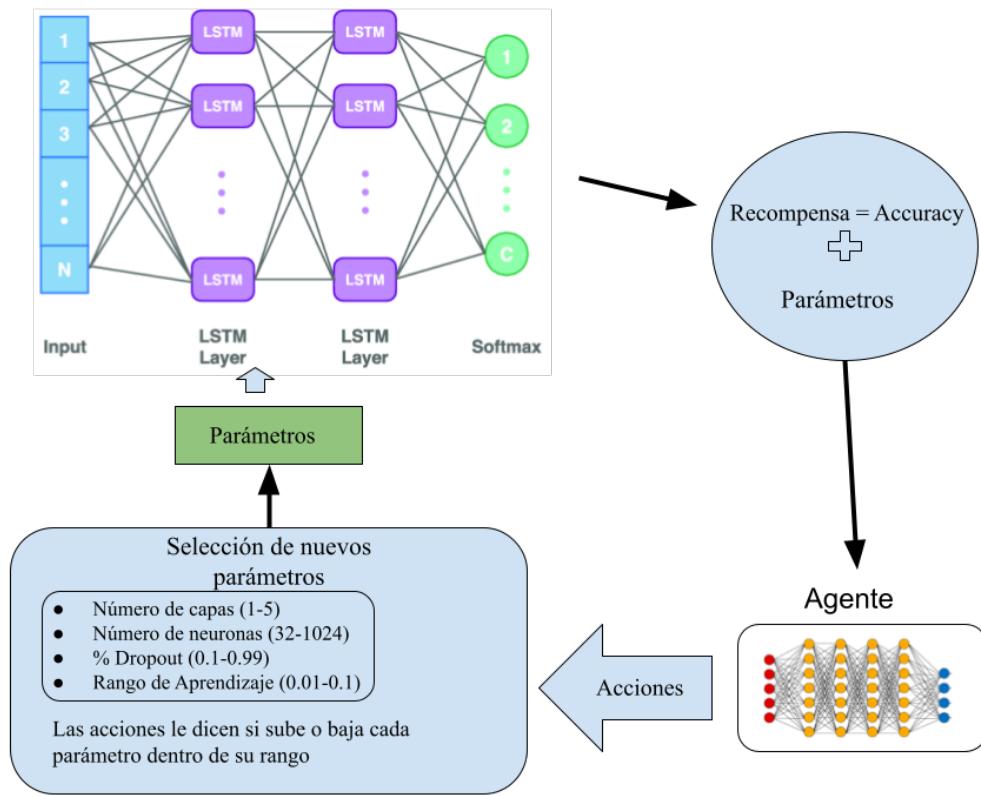


Ilustración 6.2: Diagrama de Bloques de la búsqueda de Hiperparametros para modelos basados en LSTM

Con respecto a los datos de entrada, se realizan 2 tipos de configuraciones:

- Tipo 1: Solo se entregan valores de cierre.
- Tipo 2: Se le entregan valores de apertura, cierre, máximo, mínimo y volumen junto con el promedio y la desviación estándar de las noticias.

Es importante señalar que tanto los datos de entrenamiento como los de evaluación en el proceso de optimización de hiperparametros fueron ventanas del total de activos con entrega aleatoria, ya que se busca encontrar una arquitectura que logre maximizar el nivel de acierto para un caso general.

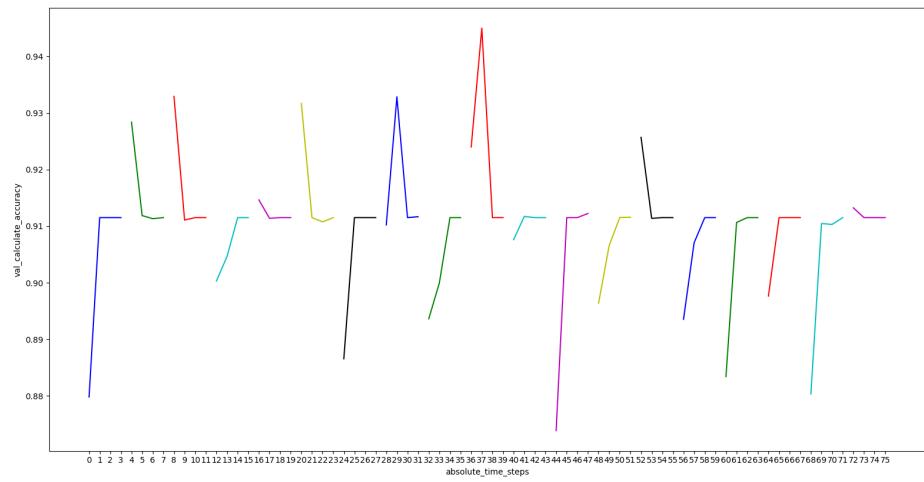


Ilustración 6.3: Proceso de Optimización de Hiperparametros en arquitectura LSTM Tipo 1.

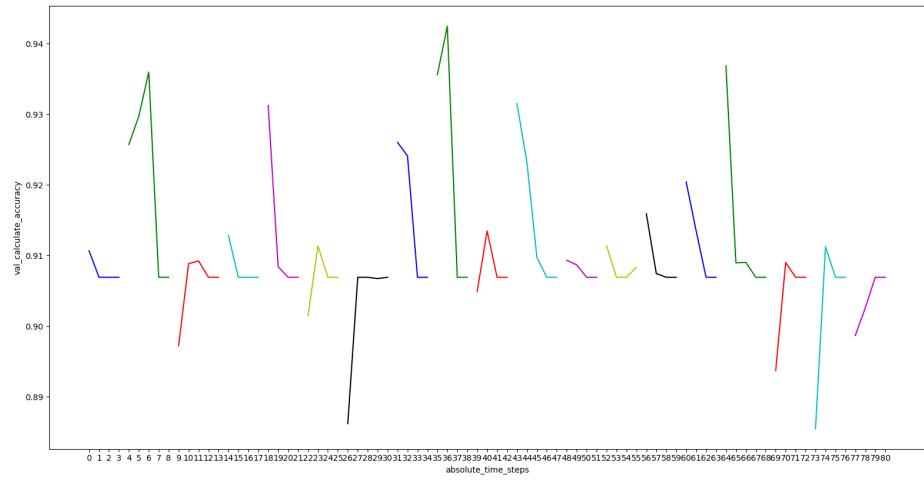


Ilustración 6.4: Proceso de Optimización de Hiperparametros en arquitectura LSTM Tipo 2.

Como se puede ver en la ilustraciones 6.3 y 6.4 , en ambos procesos el rango de acierto en la evaluación está entre 88 % y 94 % encontrando el optimo en la mitad del proceso completo. La descripción de las curvas en su mayoría ascendentes, nos permite afirmar que el método de Optimización de parámetros utilizado es útil para este tipo de modelos, ya que el agente logra identificar acciones que mejoran el comportamiento de la red en el proceso de evaluación.

Los parámetros óptimos en estos dos procesos se presentan en la tabla 6.2.

Parámetro	Tipo 1	Tipo 2
Número de capas LSTM	3	5
Número de neuronas en capas LSTM	1024	1024
Dropout en capa LSTM	0.1	0.01
Dropout recurrente en capa LSTM	0.1	0.771132
Rango de aprendizaje	0.046143	0.1
Accuracy evaluado	0.944998	0.942473

Tabla 6.2: Resultados de parámetros óptimos en modelos basados en LSTM según tipo de dato de entrada.

Como se puede ver en la tabla 6.2 , los valores de acierto promedio en ambas arquitectura son bastante similares (94 %). A partir de estas dos arquitecturas, se entrenó un modelo con cada activo en forma individual. Debido al menor número de datos no se logró obtener los resultados de acierto descrito en la tabla 6.2 obteniendo un menor porcentaje de acierto en algunos casos. Por ejemplo para el entrenamiento con los datos de “ INTC ” en la arquitectura Tipo 1 se obtuvo 84.2 % de acierto y para la de Tipo 2 un 85.51 % de acierto. En ambos casos bastante más bajo que 94 %. A continuación se presentan graficos comparativos entre estas dos arquitecturas predictivas junto con el valor de cierre real .

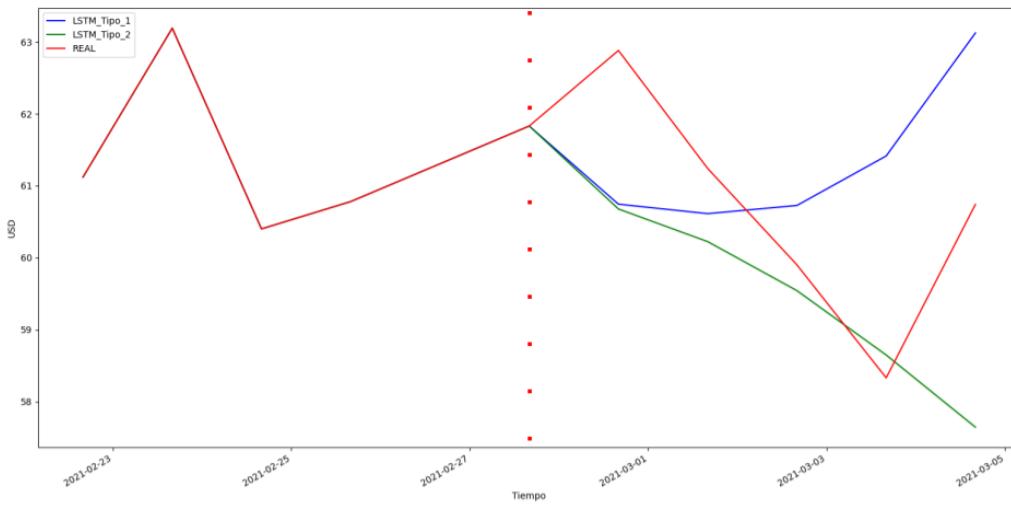


Ilustración 6.5: Gráfico Comparativo en modelos basados en LSTM ('INTC'). Ventana evaluada : [2021-02- 22/2021-02- 26]. Ventana predictiva : [2021-03-01/2021-03-05].

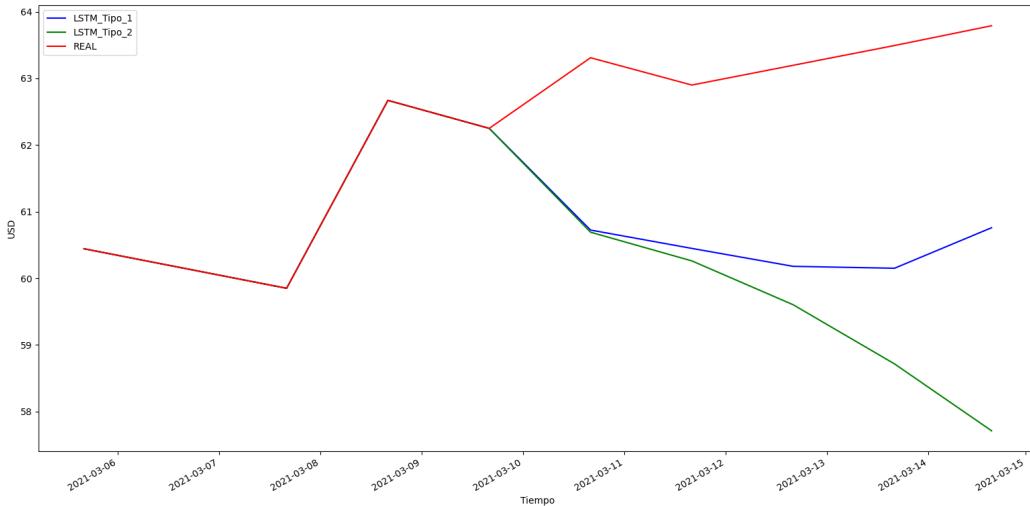


Ilustración 6.6: Gráfico Comparativo en modelos basados en LSTM ('INTC'). Ventana evaluada: [2021-03- 05/2021-03- 10]. Ventana predictiva : [2021-03-11/2021-03-15].

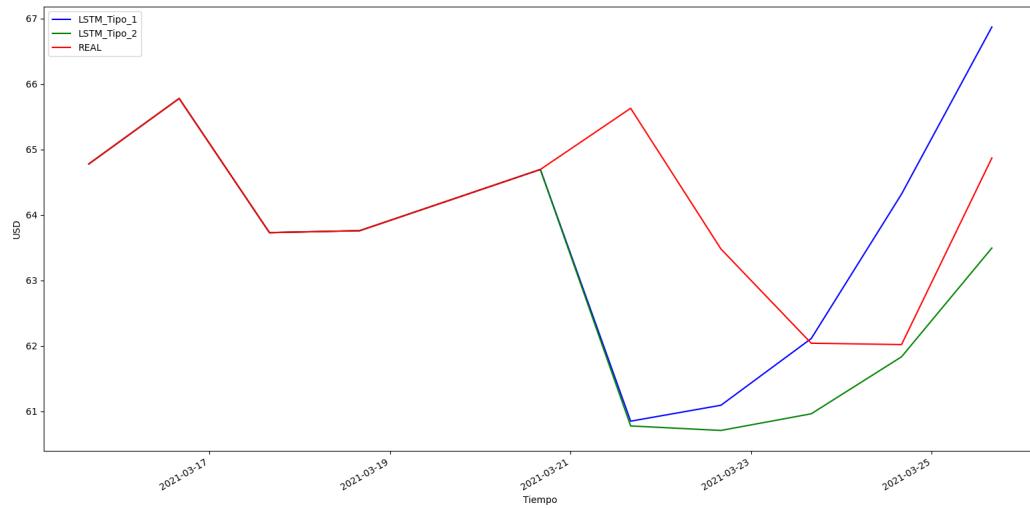


Ilustración 6.7: Gráfico Comparativo en modelos basados en LSTM ('INTC'). Ventana evaluada : [2021-03- 15/2021-03- 19]. Ventana predictiva : [2021-03-22/2021-03-26].

6.2. Modelo con Arquitectura basada en Redes Generativas Adversarias

En esta etapa el objetivo es optimizar los resultados de TimeGAN. Para esto se realizaron 3 experimentos, todos con el objetivo de optimizar la descripción distributiva de los precios de cierre del activo en una ventana de 5 días.

6.2.1. Experimento 1

El objetivo de este experimento es presentar un punto de partida. Para ello no se realizan cambios a nivel estructural del modelo original, es decir todas sus componentes (generador, discriminador, supervisor , codificador y decodificador) se construyen a partir de redes LSTM y la entrada al generador son valores aleatorios con distribución normal.

Luego de realizar el experimento 1 (Resultados en Anexo D.1), se puede concluir que si bien esta arquitectura permite describir la distribución de probabilidad no es suficiente como para generar una proyección futura aceptables.

6.2.2. Experimento 2

En el segundo experimento, se entregan valores reales de una ventana previa (los 10 factores más importantes sumado a los precios (OHLC)) como entrada al generador. No se realizan cambios en el tipo de neurona utilizada en los componentes de la arquitectura original, pero sí a nivel estructural. Para esto nuevamente se optimizó a partir de aprendizaje reforzado(Ilustración 6.8), cuya métrica de monitoreo que se busca maximizar fue la suma de la función de coste tanto del generador como la discriminador junto con el porcentaje de acierto entre datos generados y reales. Por tanto la recompensa está condicionada a este resultado en la etapa de evaluación.

Para no complicar aún más la tarea del agente, se establecen 1000 ciclos de entrenamiento y 100 ciclos de evaluación. Los parámetros considerados en el proceso de optimización de hiperparametros son descritos en la tabla 6.3.

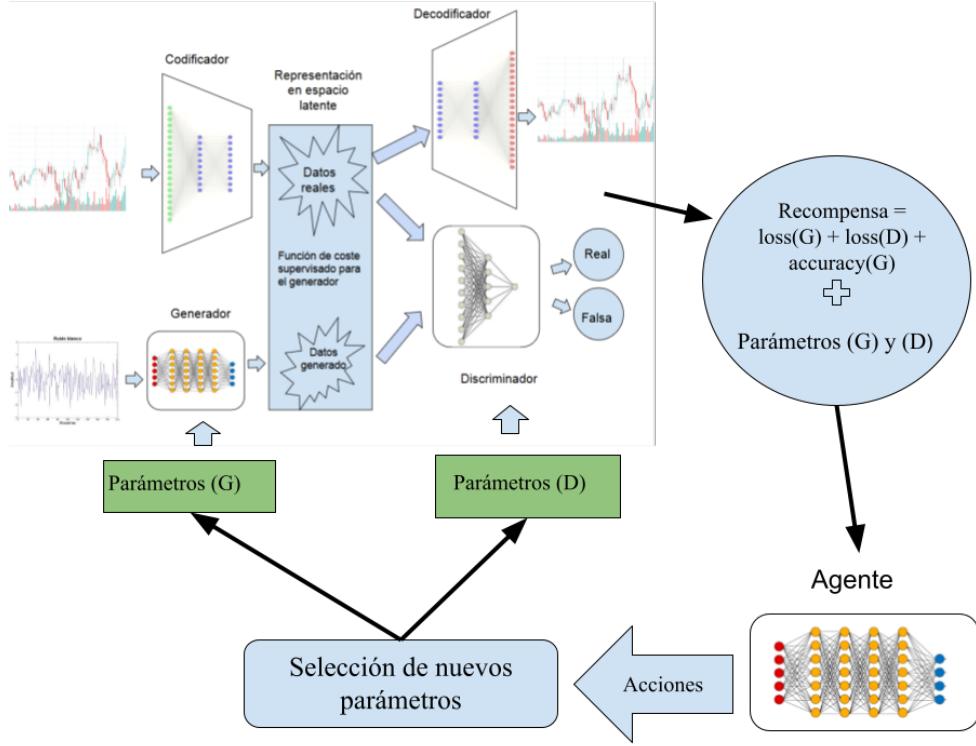


Ilustración 6.8: Diagrama de Bloques de la búsqueda de hiperparámetros para los modelos basados en TimeGAN

Parámetro	Rango	Valor Óptimo
Número de capas generador	1-5	4
Número de capas Autoencoder	1-5	1
Número de capas discriminador	1-5	5
Tamaño de batch	1-10	2
Dimensión del espacio latente	5-200	85

Tabla 6.3: Experimento 2. Tabla de parámetros óptimos y rango en el proceso de optimización para TimeGAN.

En los experimentos diseñados para generar proyecciones a partir de arquitectura Time GAN, en ningún caso el objetivo fue obtener predicciones exactas asumiendo como base el factor emocional humano presente en el comportamiento de un activo. Ese factor aleatorio se puede tratar de modelar a nivel global pero nunca definiendo un único camino, es decir predecir el futuro exacto es imposible. Es por eso que a partir de la arquitectura óptima (tabla ??), se realizaron una serie de entrenamientos aumentando el número de épocas en forma gradual para luego calcular el promedio de estas predicciones (Ilustración 6.9). De esta manera es posible obtener una información

global de las diferentes proyecciones construidas sobre el aprendizaje de la distribución de probabilidad intrínseca del activo.

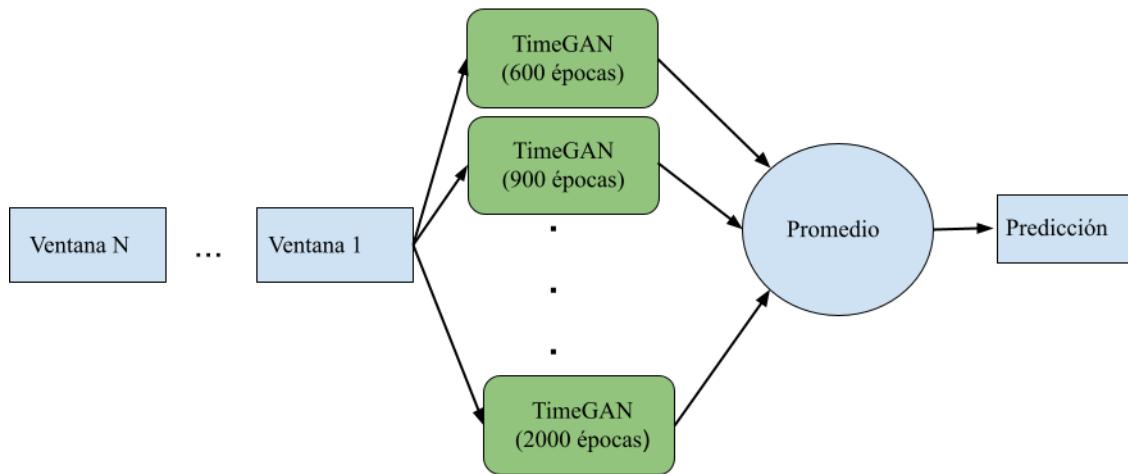


Ilustración 6.9: Diagrama metodológico del entrenamiento realizado en los modelos basados en TimeGAN.

A continuación se presentan los resultados cualitativos de este entrenamiento gradual para el caso de 'INTC'. De esta manera se presentan dos gráficos por ventana de tiempo evaluada. El primero muestra las proyecciones de todos los entrenamientos y el segundo grafica el promedio. Ambos presentan también los valores reales para comparar en forma cualitativa.

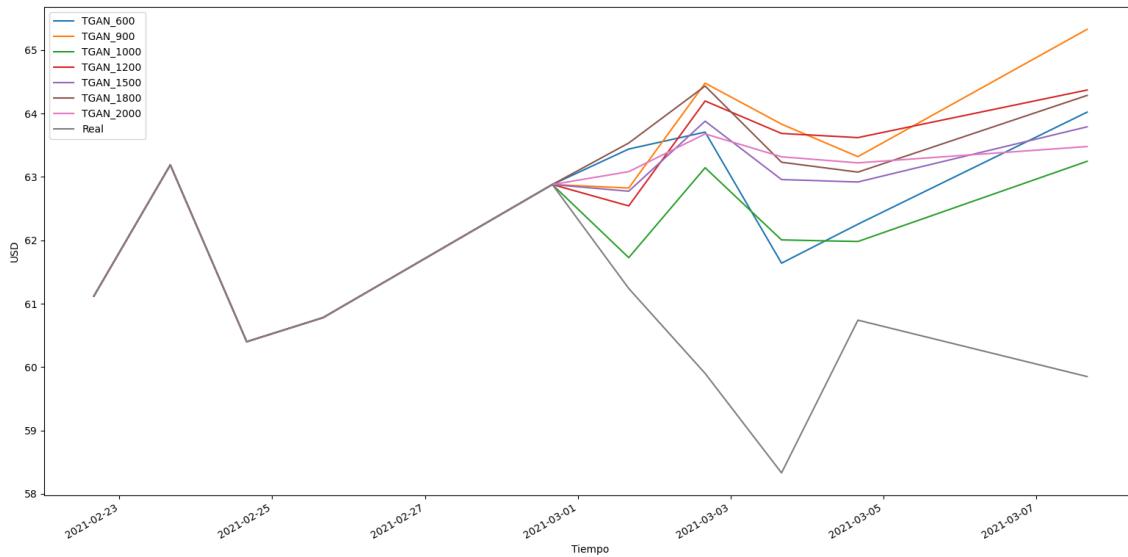


Ilustración 6.10: Experimento 2. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]

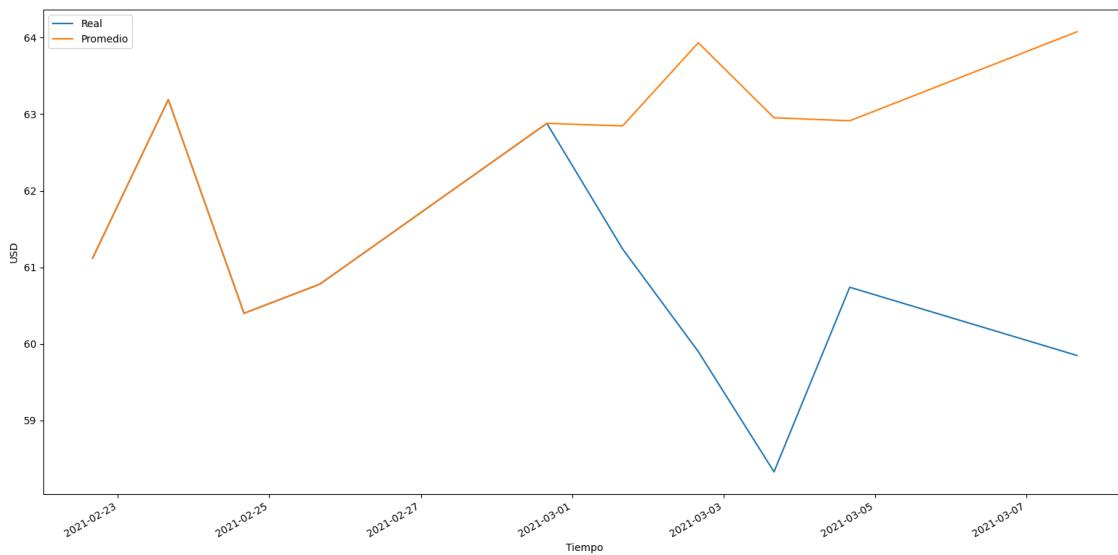


Ilustración 6.11: Experimento 2. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]

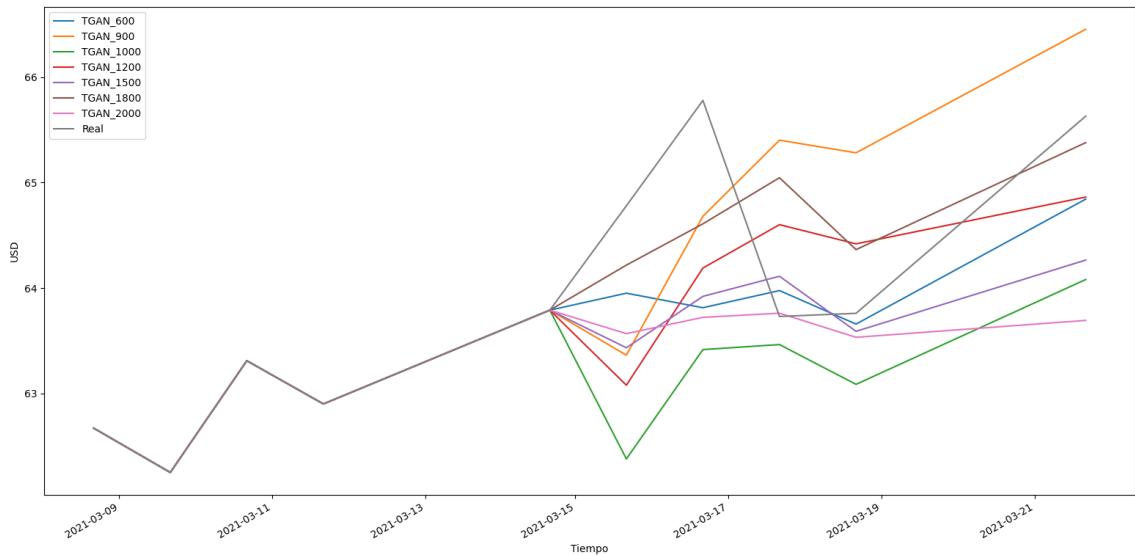


Ilustración 6.12: Experimento 2. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-9/2021-03-15]

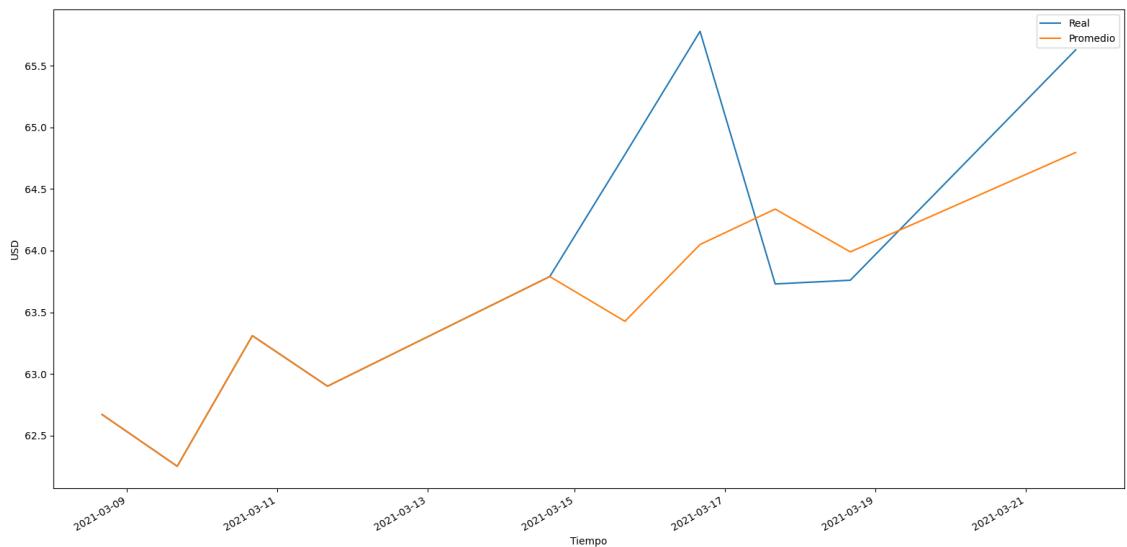


Ilustración 6.13: Experimento 2. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-9/2021-03-15]

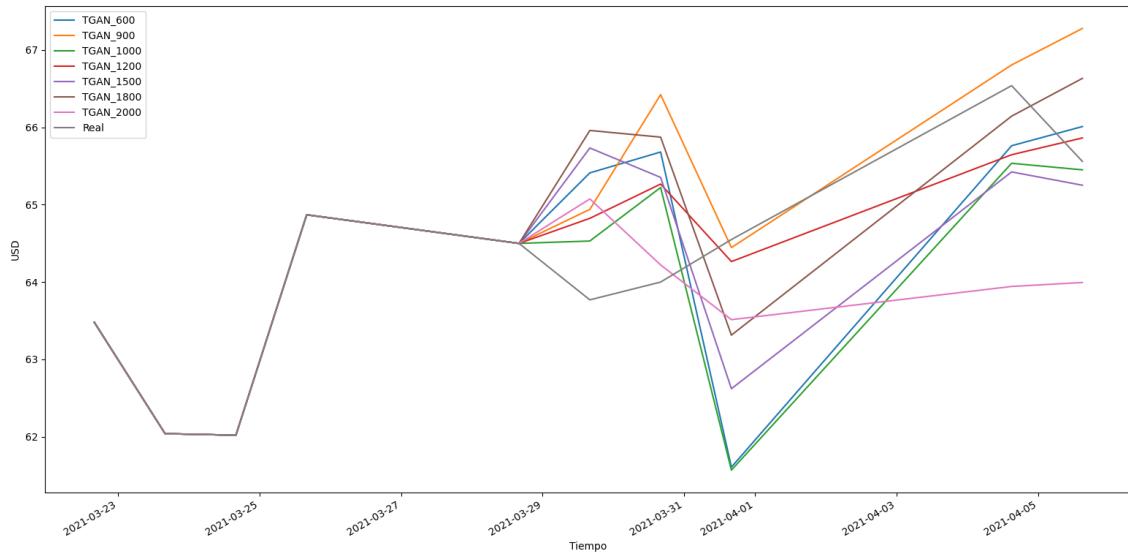


Ilustración 6.14: Experimento 2. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]

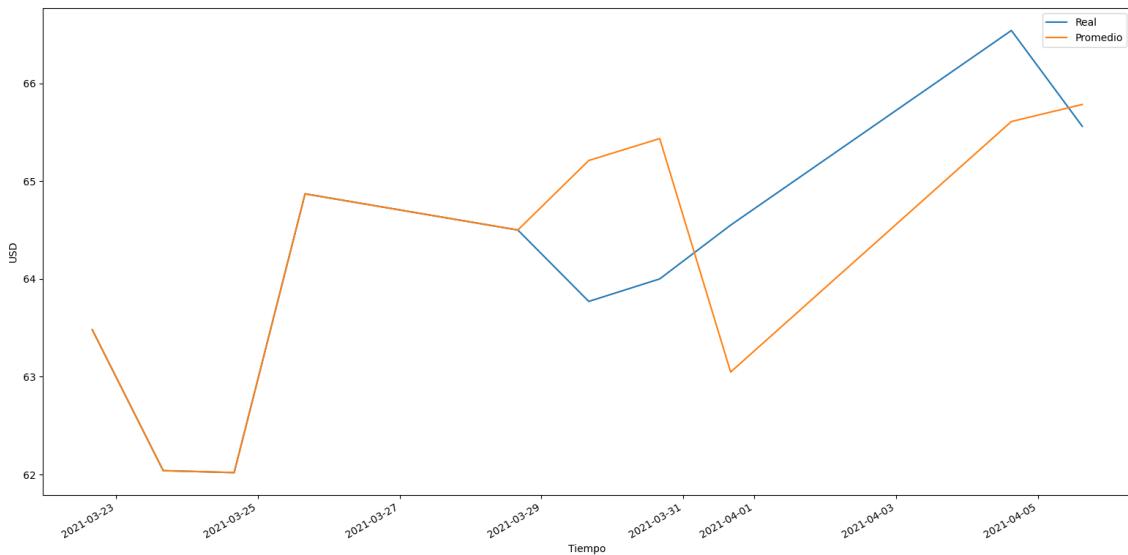


Ilustración 6.15: Experimento 2. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]

El último gráfico muestra los resultados del promedio en todo el conjunto de evaluación.

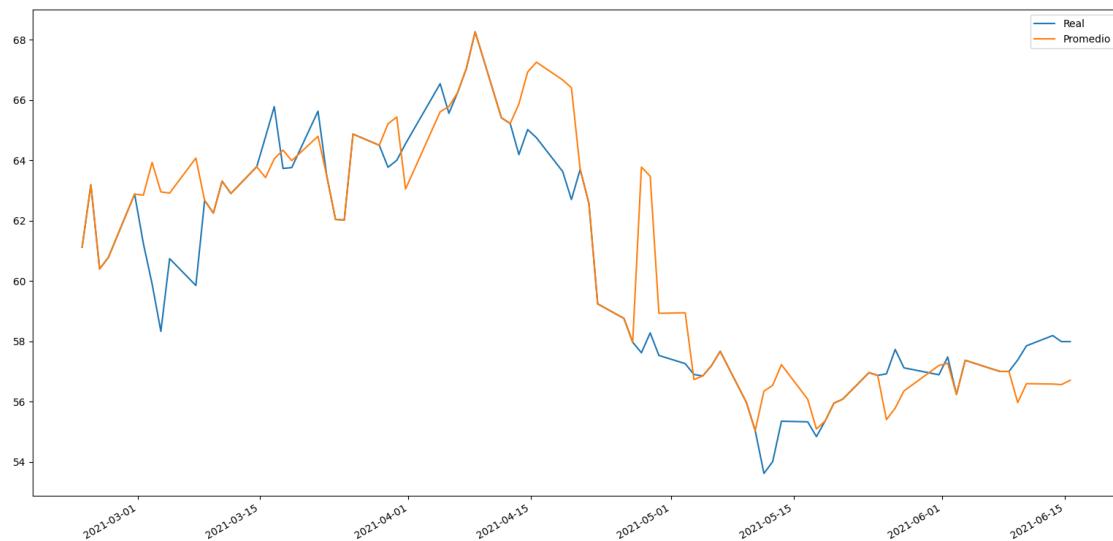


Ilustración 6.16: Experimento 2. Gráfico Comparativo Promedio ('INTC') con todo el conjunto de datos de evaluación . Conjunto de evaluación : [2021-02- 23/2021-06-15]

6.2.3. Experimento 3

En esta etapa final de experimentación con redes generativas adversarias se realizan cambios estructurales :

- a) Arquitectura del discriminador basada en neuronas convolucionales, con normalización a nivel de lote y función de activación Alfa Relu
- b) Incorporar Dropout en arquitectura del discriminador para evitar overfitting
- c) Implementar una función de rango de aprendizaje en forma triangular

Considerando estos aspectos se realiza la misma búsqueda de hiperparámetros realizada en el experimento dos. Las variables a optimizar se describen en la tabla 6.4.

Parámetro	Rango	Valor Óptimo ('INTC')
Número de capas LSTM	1-5	2
Número de neuronas en las capas LSTM	32-1024	306
Porcentaje de dropout en las capas LSTM	0.01-0.99	0.120942
Tamaño de batch	1-10	8
filter CNN	32-64	39
Alfa en Relu	0.01-0.1	0.082678
Momentum de normalización en Batch	0.01-0.99	0.023548
Learning rate máximo de CNN	0.01-2.0	1.633936
Dimensión del espacio latente	5-200	160

Tabla 6.4: Experimento 3. Tabla de parámetros óptimos y rango en el proceso de optimización para TimeGAN

A partir de esta nueva arquitectura, se realizó el mismo entrenamiento que en el experimento 2. Los resultados cualitativos no distan en demasiado a los presentados en el experimento 2, es por ello que en virtud de la longitud del documento se exponen en Anexo D.2.

6.3. Evaluación Comparativa entre Proyecciones basadas en LSTM y arquitectura TimeGAN

Por un lado, en todos los experimentos descritos anteriormente existe la evaluación cuantitativa . Para el caso de 'INTC'(tabla 6.5) los modelos basados en TimeGAN superan en promedio a los basados en LSTM por 10 % en el conjunto de evaluación.

Modelo predictivo	Accuracy Evaluacion (%)
LSTM Tipo 1	84.2
LSTM Tipo 2	85.51
TGAN (Experimento 2)	96.794138
TGAN (Experimento 3)	94.4998

Tabla 6.5: Experimento Comparativo.Tabla de evaluación cuantitativa para 'INTC'.

Por otro lado, el objetivo de utilizar arquitecturas de redes generativas adversarias no es describir un solo futuro a partir de los datos previos (como se hace con los modelos basados en LSTM), sino enseñarle al generador a describir una distribución de probabilidad característica del activo . Es por esto que la calidad de estas predicciones no se pueden medir únicamente según una métrica. Por tanto se requiere realizar una

evaluación cualitativa. Debido a la diferencia de datos utilizados por las diferentes arquitecturas, fue necesario identificar aquellos días presentes en ambos conjuntos de evaluación para así poder comparar en las mismas condiciones temporales. A continuación se presentan dos comparaciones cualitativas. Primero se evalúan todos los modelos en forma gráfica y luego se compara la utilidad de las proyecciones en un entorno financiero.

6.3.1. Comparación Cualitativo de modelos predictivos

En esta sección se presentan los resultados predictivos de todos los modelos presentados anteriormente en el activo 'INTC'. Como se puede ver a simple vista en las Ilustraciones 6.20-6.23, los resultados predictivos generados por arquitecturas TimeGAN presentan mayor cercanía a los valores reales en relación a las predicciones generadas por redes LSTM. Esto justifica los resultados de la tabla 6.5, donde la diferencia porcentual refleja la distancia promedio entre los valores reales y las predicciones.

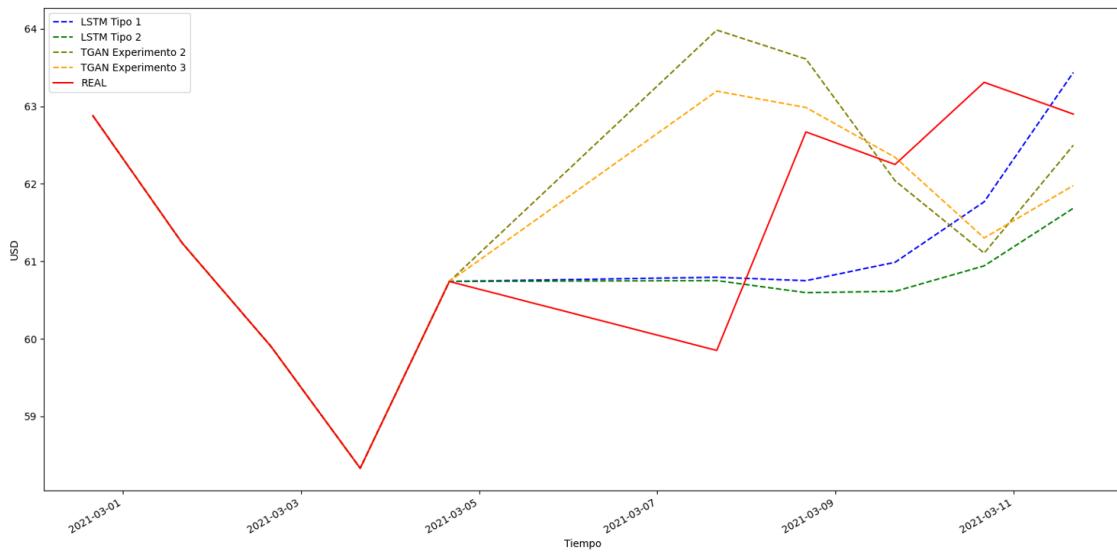


Ilustración 6.17: Experimento Comparativo. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 01/2021-03- 05]. Ventana predictiva : [2021-03-08/2021-03-12]

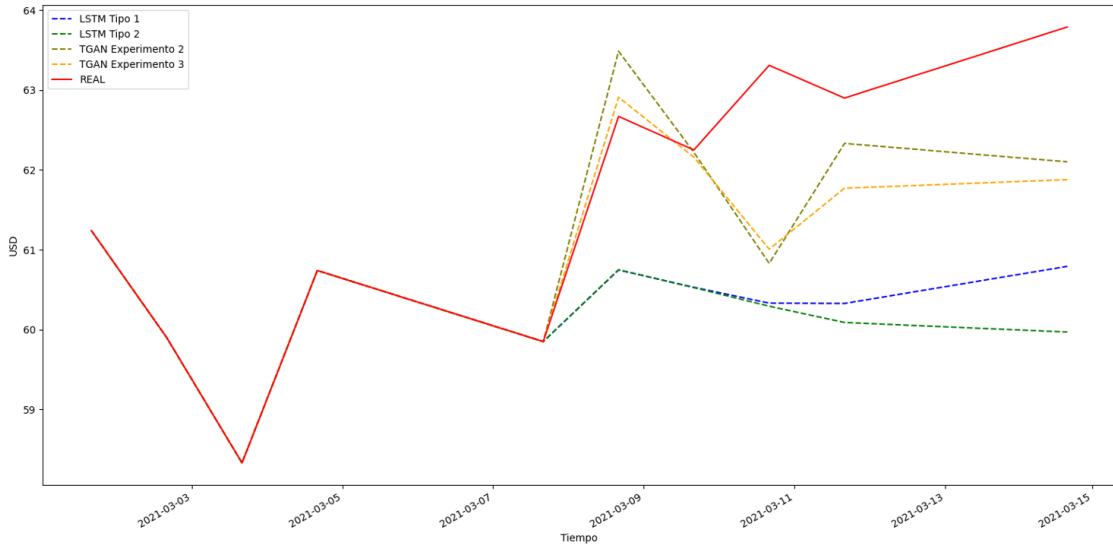


Ilustración 6.18: Experimento Comparativo. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-09/2021-03-15]

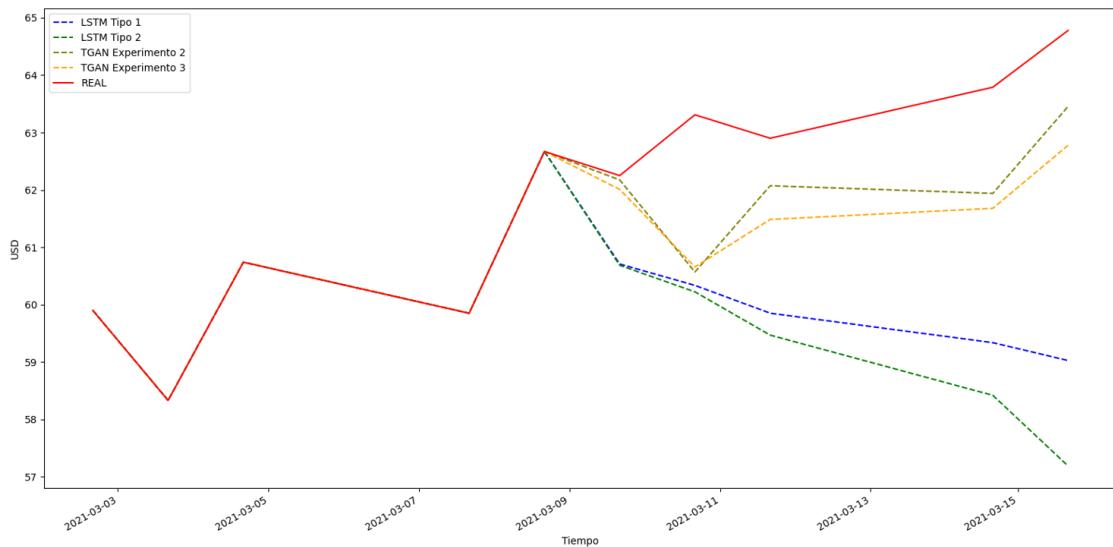


Ilustración 6.19: Experimento Comparativo. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 03/2021-03- 09]. Ventana predictiva : [2021-03-10/2021-04-16]

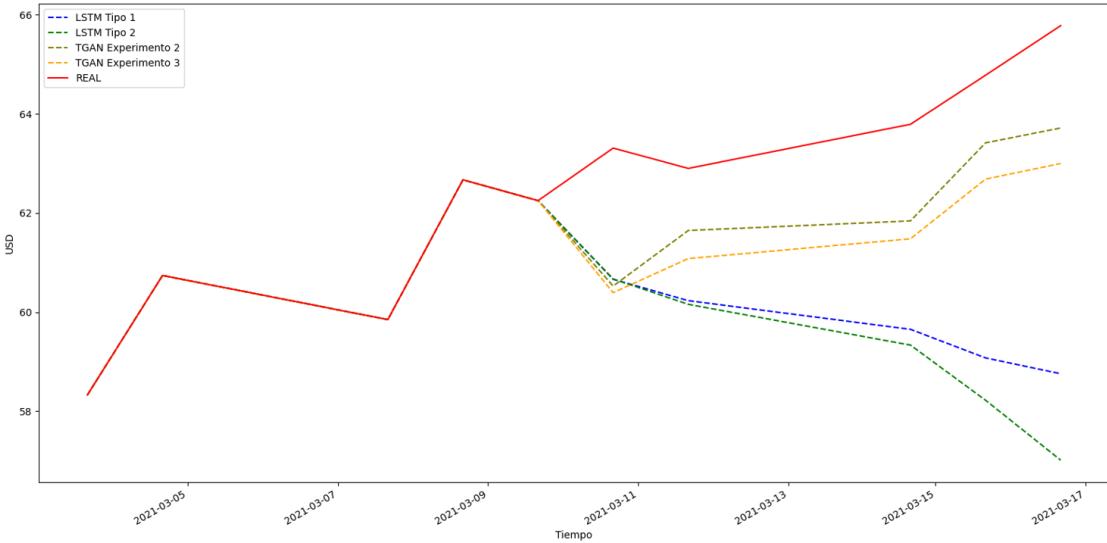


Ilustración 6.20: Experimento Comparativo. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [[2021-03- 04/2021-03- 10]. Ventana predictiva : [2021-03-11/2021-04-17]]

6.3.2. Experimento en Entorno Financiero

Se asume que al aumentar las ganancias por parte de un agente dentro de un entorno de trading mediante la incorporación de proyecciones , es probable que esta información sea útil para un analista financiero humano. Por tanto, para validar la información entregada tanto por los modelos basados en LSTM (sección 6.1) como en los modelos generativos adversarios (sección 6.2), se entrena un agente mediante aprendizaje reforzado profundo para decidir si en un instante de tiempo es apropiado comprar, vender o mantener la posición de un activo, con el objetivo de maximizar las ganancias a largo plazo. Los estados posibles están determinados a partir de una ventana de 5 días previos al día evaluado sumado a la proyección de los 5 días futuros. La recompensa del juego está asociada a las ganancias que puede obtener el agente a largo plazo.

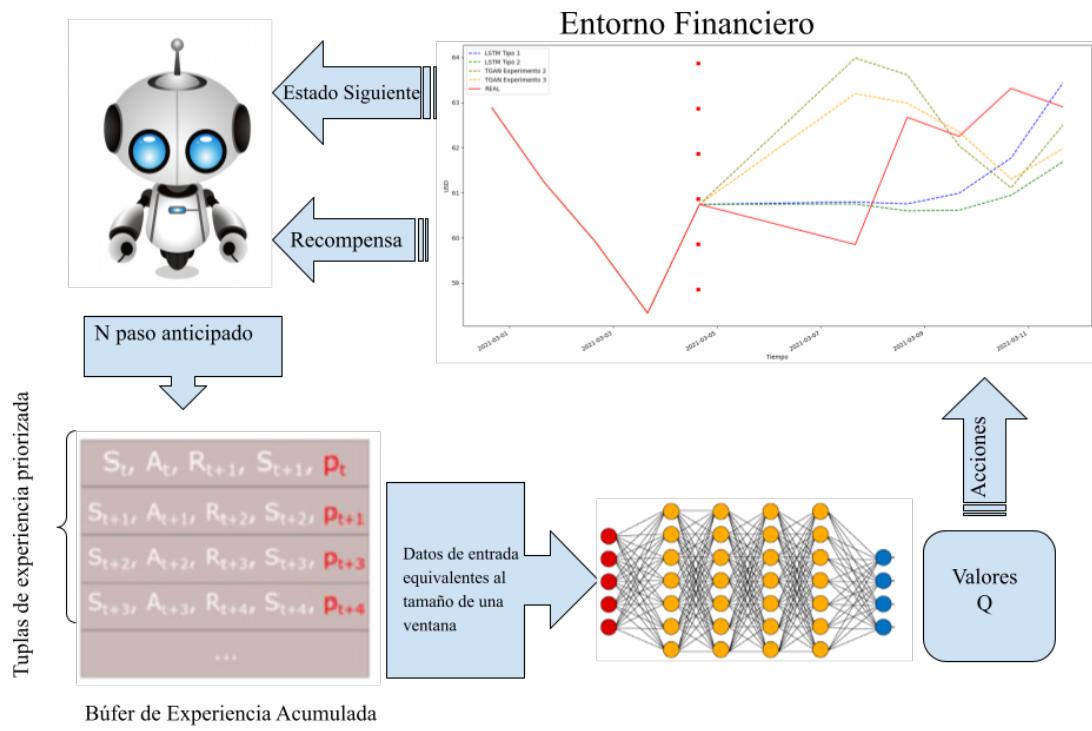


Ilustración 6.21: Experimento Comparativo en entorno Financiero (DQL)

Primero se entrena al agente sólo con ventanas pasadas y luego se realiza otro entrenamiento para que este aprenda a tomar decisiones a partir de ventanas pasadas junto con predicciones futuras. Con el segundo agente, se evalúa el desempeño con predicciones y con valores futuros reales considerando este último como una predicción con un 100 % de acierto ubicándonos en un caso utópico. La tabla 6.6 muestra las ganancias generadas por el agente en tres diferentes activos evaluados.

Activo	TGAN(Exp 2)	TGAN(Exp 3)	LSTM 1	LSTM 2	Real c/futuro	Real s/futuro
AAPL	140.160	123.360	49.510	99.150	50.909	57.690
IBM	416.730	401.380	505.539	502.610	106.239	386.219
INTC	43.510	32.730	24.899	18.549	20.460	-60.990

Tabla 6.6: Resultados de experimento en entorno financiero

Como se puede ver en la tabla 6.6, en los tres activos evaluados se logran mejores ganancias al entregar información de la siguiente ventana. Para el caso particular de AAPL el agente logra notorias mejoras con información de arquitecturas generativas,

superando con ambas arquitecturas de este tipo al mejor resultado a partir información entregada por solo redes LSTM.Por otro lado en el caso de IBM, el agente obtiene mayor retorno a partir de predicciones con arquitecturas LSTM.En todos los casos se generan buenos niveles de ganancias, esto se debe a una tendencia al alza clara en la ventana evaluada.

Al analizar el activo INTC se puede ver que debido a una clara tendencia a la baja del precio en la ventana evaluada el agente con solo información pasada no logra obtener beneficios monetarios, por el contrario pierde casi 61 dólares.Esta misma tendencia bajista se le atribuye al bajo desempeño por el agente en los casos con información futura, sin embargo es posible obtener algo de ganancia.Este factor es de suma importancia, ya que demuestra los beneficios de las predicciones en casos de que el precio se desplome.

Otro aspecto importante se puede ver en el desempeño por parte del agente en relación al caso utópico (se entregan valores futuros reales) cuyo desempeño es peor que con predicciones.Esto nos muestra la capacidad que tiene el agente de aprender a partir del factor de error intrínseco de la predicción.Esto demuestra la capacidad de interacción existente en diferentes herramientas de inteligencia artificial.

Finalmente es posible concluir que las herramientas predictivas entregan información útil en la toma de decisiones financieras.

Capítulo 7

Conclusiones

7.1. Discusión

A pesar de las complicaciones que presenta modelar el comportamiento de un activo debido a la gran cantidad de factores que afectan en él, los modelos inteligencia artificial estudiados en este trabajo permiten direccionar posibles soluciones a esta tarea. Para esto es necesario interrelacionar modelos que cubran el análisis de cada aspecto contextual relacionado al precio, siendo ésta la intención de los objetivos planteados en la sección 1.2. Cada uno de estos objetivos fueron cumplidos a lo largo de esta investigación, permitiendo recapitular las siguientes conclusiones generales . En primer lugar, la clasificación de sentimientos financieros en forma automática permite entregar a los modelos LSTM información contextual útil porque refleja un mayor porcentaje de acierto en todos los activos evaluados. Si bien no fue posible mejorar el rendimiento del clasificador original a partir del entrenamiento léxico adversario, el modelo original permitió cumplir con la tarea en forma eficiente.

En segundo lugar, las redes generativas presentan mejores resultados cualitativos en relación con los modelos LSTM luego de realizar entrenamientos con el mismo conjunto de datos sumado a los descritos en la metodología. Esto, probablemente, se produce por el entrenamiento adversario, en el cual el generador formado, también por neuronas LSTM, logra complementar su aprendizaje supervisado a partir de un aprendizaje no supervisado. La incorporación del entrenamiento no supervisado obliga al generador a encontrar relaciones en un conjunto mayor de variables.

Finalmente, el experimento en un entorno financiero presenta resultados favorables al incorporar información predictiva, lo que demuestra que ésta permite tomar mejores decisiones en la compra y venta de activos. Si bien la hipótesis de los mercados eficientes señala que el precio de un activo refleja toda la información existente, ésta

investigación permite afirmar que mediante modelos de inteligencia artificial es posible procesar información contextual útil para generar proyecciones futuras relacionadas con la toma de decisiones financieras.

7.2. Futuras Investigaciones

En este subcapítulo, se entregan algunas sugerencias fundamentales para futuras investigaciones. Estas son las siguientes:

- a) Construir un nuevo conjunto de oraciones financieras clasificadas para probar los beneficios del entrenamiento léxico adversario. También es posible adaptar tareas de Bert, como pregunta y respuesta, en el ámbito financiero.
- b) Realizar nuevos cambios estructurales en los modelos generativos como, por ejemplo, incorporar arquitectura basada en U-NET(Stoller, Tian, Ewert Dixon, 2020) con el propósito de codificar y decodificar el espacio latente.
- c) Adaptar este modelo generativo adversario utilizando computadores cuánticos,(Christa Zoufal, 2019), lo que podría presentar mejoras importantes en describir el factor aleatorio intrínseco en este tipo de datos.
- d) Construir un entrenamiento reforzado con entorno financiero realista en el cual el agente disponga de una cantidad limitada para invertir con la finalidad de que aprenda a tomar decisiones considerando esta limitante.

Bibliografía

- Araci, D. (2019). Finbert: Financial sentiment analysis with pre-trained language models.
- Ariyo, A. A., Adewumi, A. O., & Ayo, C. K. (2014). Stock price prediction using the arima model. In *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, (pp. 106–112).
- Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2019). A comparative analysis of xgboost.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.
- Fama, E. F. (1965). The behavior of stock-market prices. *The Journal of Business*, 38(1), 34–105.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- Graves, A. (2014). Generating sequences with recurrent neural networks.
- Haykin, S. S. (2009). *Neural networks and learning machines* (Third ed.). Upper Saddle River, NJ: Pearson Education.
- Hochreiter, S., Bengio, Y., Frasconi, P., & Schmidhuber, J. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In S. C. Kremer & J. F. Kolen (Eds.), *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press.
- Hochreiter, S. & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780.
- Ian J. Goodfellow, Jean Pouget-Abadie, M. M. (2014). Generative adversarial nets.
- Jiang, W. (2020). Applications of deep learning in stock market prediction: recent progress.
- Kyrychko, Y. & Hogan, S. (2010). On the use of delay equations in engineering applications. *Journal of Vibration and Control - J VIB CONTROL*, 16.

- Malkiel, B. G. (2003). The efficient market hypothesis and its critics. *Journal of Economic Perspectives*, 17(1), 59–82.
- Malo, P., Sinha, A., Takala, P., Korhonen, P., & Wallenius, J. (2013). Good debt or bad debt: Detecting semantic orientations in economic texts.
- Mohammed, M., Khan, M., & Bashier, E. (2016). *Machine Learning: Algorithms and Applications*.
- Olabe, X. B. (2008). Redes neuronales artificiales y sus aplicaciones.
- Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In Dasgupta, S. & McAllester, D. (Eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, (pp. 1310–1318)., Atlanta, Georgia, USA. PMLR.
- Romero, R. A. C. (2019). Generative adversarial network for stock market price prediction.
- Rumelhart, D., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536.
- Shah, R., Tambe, A., Bhatt, T., & Rote, U. (2020). Real-time stock market forecasting using ensemble deep learning and rainbow dqn. *EngRN: Operations Research (Topic)*.
- Sherstinsky, A. (2020). Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404, 132306.
- Siami-Namini, S. & Namin, A. S. (2018). Forecasting economics and financial time series: Arima vs. lstm.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need.
- Werbos, P. (1988). Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, 1, 339–356.
- Werbos, P. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10), 1550–1560.

WU, J., CHEN, S., & CHEN, X. (2019). Rpr-bp: A deep reinforcement learning method for automatic hyperparameter optimization. In *2019 International Joint Conference on Neural Networks (IJCNN)*, (pp. 1–8).

”Xu, J., Zhao, L., Yan, H., Zeng, Q., Liang, Y., & Sun, X. (”2019”). ”LexicalAT: Lexical-based adversarial reinforcement training for robust sentiment classification”. In *”Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* , (pp. ”5518–5527”), ”Hong Kong, China”. .^ssociation for Computational Linguistics”.

Yoon, J., Jarrett, D., & van der Schaar, M. (2019). Time-series generative adversarial networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*, volume 32, (pp. 5508–5518). Curran Associates, Inc.

Anexo A

Glosario

- **Aprendizaje Automático:** Es una disciplina del campo de la Inteligencia Artificial que, a través de algoritmos, dota a los ordenadores de la capacidad de identificar patrones en datos masivos para hacer predicciones.
- **Árbol de decisión:** El árbol de decisión es un algoritmo de aprendizaje automático no paramétrico que tiene una estructura de gráfico en forma de árbol que se utiliza para la predicción del aprendizaje automático y la hoja en un árbol de decisión corresponde a una clase objetivo y cada nodo en un árbol de decisión representa un atributo .
- **Impulso :** Impulso es un algoritmo de aprendizaje basado en conjuntos que convierte a los estudiantes de aprendizaje débil en estimadores fuertes, lo que implica entrenar modelos ML secuencialmente uno tras otro en el que en cada modelo de iteración se intenta corregir el error cometido por el modelo en la iteración anterior.
- **Transferencia de Aprendizaje :** En términos generales la transferencia de aprendizaje se refiere a la capacidad de un sistema de aplicar a una tarea nueva el conocimiento aprendido en una tarea previa . Este método de transferencia puede resultar particularmente útil cuando el problema abordado inicialmente cuenta significativamente con más datos que un segundo problema para el cual se utiliza la transferencia (Goodfellow, Bengio & Courville, 2016), permitiendo así la adición de diferentes etapas del proceso de aprendizaje en forma paulatina según las necesidades requeridas.
- **Autoencoder :** Es un tipo de red neuronal artificial que se utiliza para aprender codificaciones eficientes de datos no etiquetados. La codificación se valida y

refina al intentar regenerar la entrada a partir de la codificación.

- **Deep Learning (DL)** : Este tipo de algoritmos usan arquitectura de redes neuronales con una gran cantidad de capas.
- **Optimización de Hiperparametros** : La optimización de hiper parámetros en el aprendizaje automático tiene por objeto encontrar los hiper parámetros de un determinado algoritmo de aprendizaje automático que ofrezcan el mejor rendimiento medido en un conjunto de validación.
- **Tasa de aprendizaje (Learning Rate)** : En el aprendizaje automático y las estadísticas, la tasa de aprendizaje es un parámetro de ajuste en un algoritmo de optimización que determina el tamaño del paso en cada iteración mientras se mueve hacia un mínimo de una función de pérdida.
- **Gradiente de políticas** : Los métodos de gradiente de políticas son un tipo de técnicas de aprendizaje reforzado que se basan en la optimización de políticas parametrizadas con respecto al rendimiento esperado (recompensa acumulada a largo plazo) mediante descenso de gradiente.
- **Tokenización** : Divide cadenas de texto más largas en piezas más pequeñas o tokens. Los trozos de texto más grandes pueden ser convertidos en oraciones, las oraciones pueden ser tokenizadas en palabras, etc. El procesamiento adicional generalmente se realiza después de que una pieza de texto ha sido apropiadamente concatenada.

Anexo B

Redes Neuronales Recurrentes

B.1. Componentes Fundamentales

Como bien es explicado en (Sherstinsky, 2020), es posible derivar las componentes internas de una neurona recurrente a partir de ecuaciones diferenciales.

Sea $\vec{s}(t)$ un vector de señal de estado neuronal d-dimensional cuya evolución temporal se describe a partir de la siguiente ecuación diferencial ordinaria no homogénea de primer orden :

$$\frac{d\vec{s}(t)}{dt} = \vec{f}(t) + \vec{\phi} \quad (\text{B.1})$$

donde $\vec{f}(t)$ es una función vectorial d-dimensional dependiente del tiempo, $t \in \mathbb{R}^+$, y $\vec{\phi}$ es un vector d-dimensional constante. Una forma canónica¹ de $\vec{f}(t)$ es:

$$\vec{f}(t) = \vec{h}(\vec{s}(t), \vec{x}(t)) \quad (\text{B.2})$$

donde $\vec{x}(t)$ es el vector de señal de entrada d-dimensional y $\vec{h}(\vec{s}(t), \vec{x}(t))$ es una función vectorial cuyos argumentos también son valores vectoriales. Reemplazando, obtenemos:

$$\frac{d\vec{s}(t)}{dt} = \vec{h}(\vec{s}(t), \vec{x}(t)) + \vec{\phi} \quad (\text{B.3})$$

Las funciones de estado $\vec{s}(t)$ y de entrada $\vec{x}(t)$ utilizan esta notación vectorial ya que pueden ser funciones no solo dependientes de tiempo, t, sino también de otra variable continua independiente, $\vec{\xi}$, que denota las coordenadas en el espacio multidimensional.

¹El adjetivo canónico se usa con frecuencia en matemática para indicar que algo es natural, como debe ser e independiente de elecciones arbitrarias, que es absoluto y no relativo a un observador, que es intrínseco y no depende de un sistema de referencia o de un sistema de coordenadas, que pertenece a la estructura propia de lo que estudiamos.

nal. Usando esta notación, la intensidad de una señal de entrada de video transmitida en una pantalla plana bidimensional se representaría como $x(\vec{\xi}, t)$ con $\vec{\xi} \in \mathbb{R}^2$. El muestreo de $x(\vec{\xi}, t)$ en una cuadrícula bidimensional uniforme convierte esta señal en la representación $x(\vec{i}, t)$, donde \vec{i} es ahora un índice bidimensional discreto. Finalmente, al ensamblar los valores de $x(\vec{i}, t)$ para todas las permutaciones de los componentes del índice, \vec{i} , en un vector columna, genera $\vec{x}(t)$ como se presentó originalmente en la Ecuación B.3 anterior.

Un caso particular de $\vec{f}(t)$ en la Ecuación B.2 es:

$$\vec{f}(t) = \vec{a}(t) + \vec{b}(t) + \vec{c}(t) \quad (\text{B.4})$$

Donde $\vec{a}(t)$, $\vec{b}(t)$ y $\vec{c}(t)$, son funciones vectoriales d-dimensionales dependientes del tiempo t . La ecuación B.4 es conocida en la literatura de dinámica cerebral como “modelo aditivo”, porque agrega los términos, posiblemente no lineales, que determinan la tasa de cambio de las actividades neuronales, $\vec{s}(t)$.

Definiendo las funciones vectoriales constituyentes de la ecuación B.4 como:

$$\vec{a}(t) = \sum_{k=0}^{K_s-1} \vec{a}_k(\vec{s}(t - \tau_s(k))) \quad (\text{B.5})$$

$$\vec{b}(t) = \sum_{k=0}^{K_r-1} \vec{b}_k(\vec{r}(t - \tau_r(k))) \quad (\text{B.6})$$

$$\vec{r}(t - \tau_r(k)) = G((\vec{s}(t - \tau_r(k))) \quad (\text{B.7})$$

$$\vec{c}(t) = \sum_{k=0}^{K_x-1} \vec{c}_k(\vec{x}(t - \tau_x(k))) \quad (\text{B.8})$$

donde $\vec{r}(t)$, es una versión deformada del vector de señal de estado, $\vec{s}(t)$, denominado vector de señal de lectura y representa la salida de la neurona luego de pasar por la función de activación $G(z)$.

Entonces, el sistema resultante, obtenido al sustituir las ecuaciones B.5-B.8 en la ecuación B.4 e insertar en la ecuación 1, se convierte en:

$$\frac{d\vec{s}(t)}{d(t)} = \sum_{k=0}^{K_s-1} \vec{a}_k(\vec{s}(t - \tau_s(k))) + \sum_{k=0}^{K_r-1} \vec{b}_k(\vec{r}(t - \tau_r(k))) + \sum_{k=0}^{K_x-1} \vec{c}_k(\vec{x}(t - \tau_x(k))) + \vec{\phi} \quad (\text{B.9})$$

$$\vec{r}(t - \tau_r(k)) = G((\vec{s}(t - \tau_r(k))) \quad (\text{B.10})$$

La ecuación B.9 es una ecuación diferencial ordinario no lineal con retardos discretos (DDE)², donde la tasa de cambio en el tiempo de la señal de estado depende de tres componentes principales más el término constante (“sesgo”), $\vec{\phi}$.

- El primer componente , $\sum_{k=0}^{K_s-1} \vec{a}_k(\vec{s}(t - \tau_s(k)))$, es la combinación de hasta K_s desplazamientos en el tiempo (por las constantes de tiempo de retardo, $\tau_s(k)$ de la función , $\vec{a}_k(\vec{s}(t))$) ,la cual es en si mismo una función de señal de estado (posiblemente diferida en el tiempo) .
- El segundo componente, $\sum_{k=0}^{K_r-1} \vec{b}_k(\vec{r}(t - \tau_r(k)))$, es la combinación de hasta K_r desplazamientos en el tiempo (por las constantes de tiempo de retardo, $\tau_r(k)$ de la función, $\vec{b}_k(\vec{r}(t))$), de la señal de lectura, dada por la Ecuación B.10.
- El tercer componente, $\sum_{k=0}^{K_x-1} \vec{c}_k(\vec{x}(t - \tau_x(k)))$, que representa la entrada externa, esta formado por la combinación de hasta K_x desplazamientos en el tiempo (por las constantes de tiempo de retardo, $\tau_x(k)$) de la función, $\vec{c}_k(\vec{x}(t))$, de la señal de entrada ³.

Una opción popular para la función de deformación o activación no lineal, saturante e invertible de elementos, $G(z)$, es una forma opcionalmente escalada y/o desplazada de la tangente hiperbólica. La razón fundamental para elegir esta función de deformación es que la tangente hiperbólica posee ciertas propiedades útiles. Por un lado, es monótona y simétrica negativa con una región cuasi lineal, cuya pendiente se puede regular. Por otro lado, es bipolarmente saturante (es decir, acotado tanto en el límite negativo como en el positivo de su dominio). El modo cuasi-lineal ayuda en el diseño de los parámetros del sistema y en la interpretación de su comportamiento en el régimen de ”señales pequeñas”(es decir, $\vec{s}(t) \ll 1$). El aspecto de saturación bipolar (.”plastamiento”), junto con el diseño adecuado de los parámetros internos de las funciones $\vec{a}_k(\vec{s}(t))$ y $\vec{b}_k(\vec{r}(t))$, ayuda a mantener el estado del sistema (y, por lo tanto, su salida) limitado. El rango dinámico de las señales de estado generalmente no está restringido, pero se garantiza que las señales de lectura estarán delimitadas, mientras que aún transportan la información de estado con baja distorsión (el régimen de ”pequeña señal”) en el modo cuasi-lineal de la función de deformación . Si

²En ingeniería, los retrasos de tiempo a menudo surgen en bucles de retroalimentación que involucran sensores y actuadores (Kryuchko & Hogan, 2010)

³La señal de entrada completa, $\vec{c}(t)$, en la Ecuación B.8 a veces se denomina ”fuerza impulsora externa”(o, simplemente, la ”fuerza impulsora”) en física.

el sistema, descrito por la Ecuación B.9 y la Ecuación B.10, es estable, entonces las señales de estado también están limitadas .

Los términos de retardo de tiempo en el lado derecho de la Ecuación B.9 comprenden los aspectos de "memoria" del sistema. Permitiendo que la tasa de cambio de tiempo instantáneo de la señal de estado, $\frac{d\vec{s}(t)}{dt}$, incorpore contribuciones del estado, la lectura y los valores de la señal de entrada, medidos en diferentes puntos en el tiempo, en relación con el tiempo actual, t . Cualitativamente, estos elementos temporales enriquecen el poder expresivo del modelo al capturar información causal y / o contextual.

En las redes neuronales, el retraso de tiempo es una parte intrínseca del sistema y también uno de los factores clave que determina la dinámica ⁴.

Las funciones, $\vec{a}_k(\vec{s}(t - \tau_s(k)))$, de la señal de estado en el primer término tienen un fuerte efecto sobre la estabilidad del sistema, mientras que las funciones, $\vec{b}_k(\vec{r}(t - \tau_r(k)))$, de la señal de lectura (acotada) en el segundo término captura la mayoría de las interacciones que dan forma al comportamiento a largo plazo del sistema.

Suponga que $\vec{a}_k(\vec{s}(t - \tau_s(k)))$, $\vec{b}_k(\vec{r}(t - \tau_r(k)))$ y $\vec{c}_k(\vec{x}(t - \tau_x(k)))$ son funciones lineales de \vec{s} , \vec{r} y \vec{x} , respectivamente. Entonces la Ecuación B.9 se convierte en una DDE no lineal con coeficientes lineales de valores matriciales:

$$\frac{d\vec{s}(t)}{dt} = \sum_{k=0}^{K_s-1} A_k(\vec{s}(t - \tau_s(k))) + \sum_{k=0}^{K_r-1} B_k(\vec{r}(t - \tau_r(k))) + \sum_{k=0}^{K_x-1} C_k(\vec{x}(t - \tau_x(k))) + \vec{\phi} \quad (\text{B.11})$$

Además, si las matrices A_k , B_k y C_k son circulantes⁵ (o circulantes en bloque), entonces los términos de multiplicación matriz-vector de la Ecuación B.11 pueden expresarse como convoluciones en el espacio de los elementos de \vec{s} , \vec{r} , \vec{x} , y $\vec{\phi}$, cada uno indexado por \vec{i} :

$$\frac{d\vec{s}(\vec{i}, t)}{dt} = \sum_{k=0}^{K_s-1} \vec{a}_k(\vec{i}) * (\vec{s}(\vec{i}, t - \tau_s(k))) + \sum_{k=0}^{K_r-1} \vec{b}_k(\vec{i}) * (\vec{r}(\vec{i}, t - \tau_r(k))) + \sum_{k=0}^{K_x-1} \vec{c}_k(\vec{i}) * (\vec{x}(\vec{i}, t - \tau_x(k))) + \vec{\phi}(\vec{i}) \quad (\text{B.12})$$

El índice, \vec{i} , es 1-dimensional si las matrices, A_k , B_k y C_k , son circulantes y multidimensionales si son circulantes en bloque ⁶.

⁴En las redes neuronales, se produce un retraso de tiempo en la interacción entre neuronas; es inducida por la velocidad de commutación finita de la neurona y el tiempo de comunicación entre neuronas.

⁵Cada columna se obtiene de la anterior al hacer un desplazamiento cíclico hacia abajo.

⁶Por ejemplo, la forma bidimensional de \vec{i} es apropiada para tareas de procesamiento de imágenes

Las sumas de términos retardados en el tiempo en la Ecuación B.12 representan convoluciones en el dominio del tiempo con núcleos de tamaño finito, que consisten en las convoluciones espaciales $\vec{a}_k(\vec{i}) * \vec{s}(\vec{i})$, $\vec{b}_k(\vec{i}) * \vec{r}(\vec{i})$, y $\vec{c}_k(\vec{i}) * \vec{x}(\vec{i})$) como los coeficientes para los tres componentes temporales, respectivamente. De hecho, si todo el conjunto de datos (por ejemplo, el conjunto de datos de entrada, $\vec{x}(t)$) está disponible a priori para todo el tiempo antes de la aplicación de la Ecuación B.12, entonces algunos de los retrasos de tiempo correspondientes (por ejemplo, $\tau_x(k)$) puede ser negativo, al permitir la incorporación de información “futura” para calcular el estado del sistema en el momento presente, t. Esto será relevante más adelante en el análisis.

Para el análisis correspondiente a la circulación de información de un paso nos permitimos asumir las siguientes simplificaciones :

$$\begin{aligned}
K_s &= 1 \\
\tau_s(0) &= 0 \\
A_0 &= A \\
K_r &= 1 \\
\tau_r(0) &= \tau_0 \\
B_0 &= B \\
K_x &= 1 \\
\tau_x(0) &= 1 \\
C_0 &= 0
\end{aligned} \tag{B.13}$$

La Ecuación B.11 lo convierte en:

$$\frac{d\vec{s}(t)}{dt} = A\vec{s}(t) + B\vec{r}(t - \tau_0) + C\vec{x}(t) + \vec{\phi} \tag{B.14}$$

La Ecuación B.11, la Ecuación B.12 y, por tanto, la Ecuación B.14 son Ecuaciones diferenciales con retardo DDE no homogéneas de primer orden no lineales. Una técnica numérica estándar para evaluar estas ecuaciones, o, de hecho, cualquier realización de la Ecuación B.1, es discretizadas en el tiempo y calcular los valores de las señales de entrada y las señales de estado en cada muestreo de tiempo hasta la duración total requerida, realizando integración numérica.

Al denotar la duración del paso de tiempo de muestreo como ΔT y el índice de la muestra de tiempo como n.

$$t = n\Delta T \quad (\text{B.15})$$

$$\frac{d\vec{s}(t)}{dt} \approx \frac{\vec{s}(n\Delta T + \Delta T) - \vec{s}(n\Delta T)}{\Delta T} \quad (\text{B.16})$$

aplicando la regla de discretización de Euler hacia atrás ⁷a la Ecuación B.14, se obtiene ⁸:

$$\frac{\vec{s}(n\Delta T + \Delta T) - \vec{s}(n\Delta T)}{\Delta T} \approx A\vec{s}(n\Delta T + \Delta T) + B\vec{r}(n\Delta T + \Delta T - \tau_0) + C\vec{x}(n\Delta T + \Delta T) + \vec{\phi} \quad (\text{B.17})$$

Ahora establecemos el retardo, τ_0 , igual al paso de tiempo único. Esto se puede interpretar como almacenar el valor de la señal de lectura en la memoria en cada paso de tiempo para ser utilizado en las ecuaciones anteriores en el siguiente paso de tiempo. Después de un solo uso, el almacenamiento de la memoria se puede sobrescribir con el valor actualizado de la señal de lectura que se usará en el siguiente paso de tiempo, y así sucesivamente ⁹. Por lo tanto, estableciendo $\tau_0 = \Delta T$ y reemplazando el signo de aproximación con un signo igual para la conveniencia en la Ecuación B.17 da:

$$\frac{\vec{s}(n\Delta T + \Delta T) - \vec{s}(n\Delta T)}{\Delta T} = A\vec{s}(n\Delta T + \Delta T) + B\vec{r}(n\Delta T) + C\vec{x}(n\Delta T + \Delta T) + \vec{\phi} \quad (\text{B.18})$$

$$\frac{\vec{s}((n+1)\Delta T) - \vec{s}(n\Delta T)}{\Delta T} = A\vec{s}((n+1)\Delta T) + B\vec{r}(n\Delta T) + C\vec{x}((n+1)\Delta T) + \vec{\phi} \quad (\text{B.19})$$

$$\vec{s}((n+1)\Delta T) - \vec{s}(n\Delta T) = \Delta T(A\vec{s}((n+1)\Delta T) + B\vec{r}(n\Delta T) + C\vec{x}((n+1)\Delta T) + \vec{\phi}) \quad (\text{B.20})$$

⁷El método de Euler hacia atrás es una regla de discretización estable que se utiliza para resolver numéricamente ecuaciones diferenciales ordinarias. Una forma sencilla de expresar esta regla es sustituir la fórmula de diferencias finitas hacia adelante en la definición de la derivada, relajar el requisito $\Delta T \rightarrow 0$ y evaluar la función en el lado derecho (es decir, la cantidad a la que es igual la derivada) en el momento, $t + \Delta T$.

⁸Es sencillo extender la aplicación de la regla de discretización a la Ecuación B.11 completa, que contiene cualquiera o todos los términos de retardo de tiempo y sus coeficientes de matriz correspondientes, sin las simplificaciones anteriores. Entonces no hay pérdida de generalidad.

⁹Nuevamente, los términos adicionales, que contienen señales de entrada retardadas en el tiempo combinadas de manera similar (como se demostrará que es beneficioso más adelante en este documento) y señales de estado, pueden incluirse en la discretización, relajando las simplificaciones anteriores según sea necesario para adaptarse a los requisitos del problema en cuestión.

Después de realizar la discretización, todas las mediciones de tiempo en la Ecuación B.20 se convierten en múltiplos integrales del paso de tiempo de muestreo, ΔT . Ahora, ΔT se puede eliminar de los argumentos, lo que deja el eje del tiempo adimensional. Por lo tanto, todas las señales se transforman en secuencias, cuyo dominio es el índice discreto, n , y la Ecuación B.14 se convierte en una ecuación en diferencias no homogénea de primer orden no lineal:

$$\begin{aligned}\vec{s}[n+1] - \vec{s}[n] &= \Delta T(A\vec{s}[n+1] + B\vec{r}[n] + C\vec{x}[n+1] + \vec{\phi}) \\ \vec{s}[n+1] &= \vec{s}[n] + \Delta T(A\vec{s}[n+1] + B\vec{r}[n] + C\vec{x}[n+1] + \vec{\phi})\end{aligned}\quad (\text{B.21})$$

$$(I - \Delta T A)\vec{s}[n+1] = \vec{s}[n] + (\Delta T B)\vec{r}[n] + (\Delta T C)\vec{x}[n+1] + (\Delta T)(\vec{\phi}) \quad (\text{B.22})$$

Definiendo:

$$W_s = \frac{1}{(I - \Delta T A)} \quad (\text{B.23})$$

y multiplicar ambos lados de la Ecuación B.22 por W_s conduce a:

$$\vec{s}[n+1] = \vec{s}[n](W_s) + (\Delta T W_s B)\vec{r}[n] + (\Delta T W_s C)\vec{x}[n+1] + \Delta T W_s \vec{\phi} \quad (\text{B.24})$$

que después de desplazar el índice, n , hacia adelante en 1 paso se convierte en:

$$\begin{aligned}\vec{s}[n] &= \vec{s}[n-1](W_s) + (\Delta T W_s B)\vec{r}[n-1] + (\Delta T W_s C)\vec{x}[n] + (\Delta T W_s)\vec{\phi} \\ \vec{r}[n] &= G(\vec{s}[n])\end{aligned}\quad (\text{B.25})$$

Definiendo dos matrices de ponderaciones adicionales y un vector de sesgo,

$$W_r = \Delta T W_s B \quad (\text{B.26})$$

$$W_x = \Delta T W_s C \quad (\text{B.27})$$

$$\vec{\theta}_s = \Delta T W_s \vec{\phi} \quad (\text{B.28})$$

transforma el sistema anterior en la forma canónica Recurrent Neural Network (RNN):

$$\vec{s}[n] = W_s \vec{s}[n-1] + W_r \vec{r}[n-1] + W_x \vec{x}[n] + \vec{\theta}_s \quad (\text{B.29})$$

$$\vec{r}[n] = G(\vec{s}[n]) \quad (\text{B.30})$$

Para que el sistema de la Ecuación B.26 sea estable, cada valor propio de $\hat{W} = W_s + W_r$ debe estar dentro del círculo unitario de valores complejos [1, 65]. Dado que existe una flexibilidad considerable en la elección de los elementos de A y B para satisfacer este requisito, es aceptable establecer $\Delta T = 1$ por simplicidad. Como otra simplificación, sea A una matriz diagonal con entradas muy negativas (es decir, $a_{ii} \ll 0$) en su diagonal principal (garantizando prácticamente la estabilidad de la Ecuación B.14). Entonces, de la Ecuación B.23, $W_s \approx A^{-1}$ será una matriz diagonal con entradas positivas, $\frac{1}{a_{ii}}$, en su diagonal principal, lo que significa que el efecto explícito del valor de la señal de estado de memoria, $\vec{s}[n-1]$, en la trayectoria del sistema será insignificante (el efecto implícito a través de $\vec{r}[n-1]$ seguirá presente mientras $\|W_r\| \geq 0$). Por lo tanto, ignorar el primer término en la Ecuación B.29, lo reduce a la definición estándar RNN:

$$\vec{s}[n] = W_r \vec{r}[n-1] + W_x \vec{x}[n] + \vec{\theta}_s \quad (\text{B.31})$$

$$\vec{r}[n] = G(\vec{s}[n]) \quad (\text{B.32})$$

Como forma abreviada, escribiremos todos los parámetros del sistema RNN estándar en la Ecuación B.31 bajo un símbolo, \ominus :

$$\ominus \equiv \left\{ W_r, W_x, \vec{\theta}_s \right\} \quad (\text{B.33})$$

De la Ecuación B.31, ahora solo la matriz $\hat{W} \approx W_r \approx -A^{-1}B$ es responsable de la estabilidad del RNN. Considere el mejor escenario de caso, donde B es una matriz simétrica ($B = B^T$). Con esta simplificación, la matriz esencial para analizar la estabilidad de la Ecuación B.31 se convierte en $\hat{W} = -[(V_B^T A^{-1})(V_B \Lambda_B)]$, donde V_B es la matriz ortogonal de los autovectores de B, y Λ_B es la matriz diagonal de los autovalores de B (con los valores propios individuales, λ_i , en la diagonal principal de Λ_B). Dado que A es diagonal y V_B es ortogonal, \hat{W} es una matriz diagonal con las entradas $\mu_i = \lambda_i a_{ii}$ en su diagonal principal. Estas cantidades se convierten en los valores propios del sistema RNN general en la Ecuación B.31 en el régimen de señal pequeña” ($\vec{s}[n] \ll 1$) cada uno sumando el modo de μ_i^n multiplicado por su condición inicial correspondiente, para la trayectoria de $\vec{s}[n]$.

Una condición necesaria y suficiente para la estabilidad es que $0 < \mu_i < 1$, lo que significa que cada valor propio, λ_i , de B debe satisfacer la condición $0 < \lambda_i < a_{ii}$. Si cualquier μ_i y λ_i no satisface esta condición, el sistema será inestable, lo que hará que

los elementos de $\vec{r}[n]$ oscilen o se saturen (es decir, ingresen a las regiones planas de la no linealidad de deformación) en algún valor del índice, n.

Una alternativa a elegir la forma conveniente específica de A en la Ecuación B.23 sería tratar (algo arbitrariamente) W_s , W_r , W_x y $\vec{\theta}_s$ en la Ecuación B.29 como parámetros mutuamente independientes y luego establecer $W_s = 0$ para obtener la definición RNN estándar (como en la Ecuación B.31). En este caso, se sigue aplicando el análisis de estabilidad anterior. En particular, los valores propios, μ_i , de W_r están sujetos al mismo requisito, $0 < \mu_i < 1$, como condición necesaria y suficiente para la estabilidad.

Es conveniente utilizar el término celda cuando se hace referencia a la Ecuación B.29 y la Ecuación B.31 en el estado no inicializado. En otras palabras, la secuencia ha sido definida por estas ecuaciones, pero sus términos aún no se han calculado. Entonces se puede decir que la celda está "desplegada." "desenrollada." especificando las condiciones iniciales en la señal de estado, $\vec{s}[n]$, y evaluando numéricamente la Ecuación 30 o la Ecuación 32 para un rango finito de pasos discretos, indexados por n . Este proceso se ilustra en el lado derecho de la igualdad en 2.3.

Tanto la Ecuación B.29 como la Ecuación B.31 son recursivas en la señal de estado, $\vec{s}[n]$. Por lo tanto, debido a la aplicación repetida de la relación de recurrencia como parte del desenrollado, la señal de estado, $\vec{s}[n]$, en algún valor del índice, n, no importa cuán grande sea, abarca las contribuciones de la señal de estado, $\vec{s}[k]$, y la señal de entrada, $\vec{x}[k]$, para todos los índices, $k < n$, terminando en k = 0 siendo este el comienzo de la secuencia . Debido a este atributo, el RNN pertenece a la categoría de los sistemas de "Respuesta de impulso infinito" (IIR).

En la práctica, es deseable aproximar una secuencia con un soporte infinito (IIR), como la Ecuación B.29 o la Ecuación B.31, mediante una secuencia de "Respuesta de impulso finito" (FIR). La razón es que los sistemas FIR tienen ciertas ventajas sobre los sistemas IIR.

- Los sistemas FIR presentan estabilidad garantizada,es decir son intrínsecamente estables
- Los sistemas FIR se pueden realizar con recursos computacionales finitos. Un sistema FIR tomará un número finito de pasos para calcular la salida de la entrada y requerirá un número finito de ubicaciones de memoria para almacenar resultados intermedios.
- La complejidad computacional y los requisitos de almacenamiento de un sistema FIR se conocen en el momento del diseño.

Para esto en (Sherstinsky, 2020) se obtiene la formulación del sistema RNN, desenrollado para K_m pasos:

$$\vec{s}[n = -1] = \vec{0} \quad (\text{B.34})$$

$$\vec{s}[n] = \begin{cases} \vec{W}_r \vec{r}[n-1] + W_x \vec{x}_m[n] + \vec{\theta}_s, & 0 \leq n \leq K_m - 1 \\ \vec{0}, & \text{otherwise} \end{cases} \quad (\text{B.35})$$

$$\vec{r}[n] = \begin{cases} G(\vec{s}[n]), & 0 \leq n \leq K_m - 1 \\ \vec{0}, & \text{otherwise} \end{cases} \quad (\text{B.36})$$

$$\vec{x}_m[n] = \begin{cases} \vec{x}[n + j(m)], & 0 \leq n \leq K_m - 1 \\ \vec{0}, & \text{otherwise} \end{cases} \quad (\text{B.37})$$

$$j(m) = \begin{cases} \sum_{i=0}^{m-1} \vec{K}_i, & 0 \leq m \leq M - 1 \\ 0, & m = 0 \end{cases} \quad (\text{B.38})$$

B.1.1. Dificultades de entrenamiento

Como algoritmo de entrenamiento supervisado, BPTT utiliza los pares de datos $\vec{x}_m[n]$ y $\vec{r}[n]$ disponibles (o los pares respectivos de algunas asignaciones de estas cantidades) en el conjunto de entrenamiento para calcular los parámetros del sistema y así optimizar una función objetivo, E, que depende de la señal de lectura, $\vec{r}[n]$, en uno o más valores del índice, n. Si se usa el desenso del gradiente (u otro algoritmo de “tipo de gradiente”) para optimizar E, entonces BPTT proporciona un procedimiento consistente para derivar los elementos de E mediante una aplicación repetida de la regla de la cadena ¹⁰

Suponiendo que la función objetivo, E, toma la misma forma para todos los segmentos. Apliquemos ahora BPTT a la Ecuación B.34. Suponga que E depende de la señal de lectura, $\vec{r}[n]$, en algún valor específico del índice, n. Entonces es razonable desear medir el gradiente total de E con respecto a $\vec{r}[n]$:

$$\vec{\chi} \equiv \nabla_{\vec{r}[n]} E = \frac{\partial E}{\partial \vec{r}[n]} \quad (\text{B.39})$$

¹⁰El nombre Retropropagación a través del tiemporefleja los orígenes de las redes neuronales recurrentes en el dominio de tiempo continuo y ecuaciones diferenciales. Si bien no es estrictamente preciso, dada la naturaleza discreta del sistema RNN en consideración, la Retropropagación a través del tiempo. ^{es}fácil de recordar , tiene un significado histórico y no debe ser fuente de confusión.

Dado que $\vec{r}[n]$ depende explícitamente de $\vec{s}[n]$, se deduce que $\vec{s}[n]$ también influye en E, y uno debería estar interesado en medir el gradiente total de E con respecto a $\vec{s}[n]$:

$$\vec{\psi}[n] \equiv \nabla_{\vec{s}[n]} E = \frac{\partial E}{\partial \vec{s}[n]} \quad (\text{B.40})$$

Muy a menudo, en la práctica, la función objetivo general se define como la suma de contribuciones separadas que involucran la señal de lectura, $\vec{r}[n]$, en cada valor individual del índice, n:

$$E = \sum_{i=0}^{K_m-1} E(\vec{r}[n]) \quad (\text{B.41})$$

Debido a la presencia de términos de penalización individuales, $E(\vec{r}[n])$, en la Ecuación B.40 para la función objetivo general del sistema, puede ser tentador usar la regla de la cadena directamente con respecto a $E([n])$ en aislamiento y simplemente concluir que $\vec{\chi}[n]$ en la Ecuación B.38 es igual a $\frac{\partial E(\vec{r}[n])}{\partial \vec{r}[n]} \odot \frac{\partial G_d(\vec{z})}{\partial \vec{z}}$, donde el operador \odot denota el producto vectorial por elementos. Sin embargo, esto perdería un componente adicional importante del gradiente con respecto a la señal de estado. La sutileza es que para un RNN, la señal de estado, $\vec{s}[n]$, en $n = k$ también influye en la señal de estado, $\vec{s}[n]$, en $n = k + 1$. La dependencia de $\vec{s}[n+1]$ en $\vec{s}[n]$ a $\vec{r}[n]$ se hace evidente al reescribir la Ecuación B.34 en el índice, $n + 1$:

$$\begin{aligned} \vec{s}[n+1] &= W_r \vec{r}[n] + W_x \vec{x}_m[n+1] + \vec{\theta}_s \\ \vec{r}[n] &= G(\vec{s}[n]) \\ \vec{r}[n+1] &= G(\vec{s}[n+1]) \end{aligned} \quad (\text{B.42})$$

Por lo tanto, teniendo en cuenta ambas dependencias, mientras se aplica la regla de la cadena, se obtienen las expresiones para la derivada parcial total de la función objetivo con respecto a la señal de lectura y la señal de estado en el índice, n:

$$\chi[n] = \frac{\partial E(\vec{r}[n])}{\partial \vec{r}[n]} + W_r \vec{\psi}[n+1] \quad (\text{B.43})$$

$$\vec{\psi}[n] = \vec{\chi}[n] \odot \frac{\partial G(\vec{z})}{\partial z} \Big|_{z=\vec{s}[n]} \quad (\text{B.44})$$

$$= \left(\frac{\partial E(\vec{r}[n])}{\partial \vec{r}[n]} + W_r \vec{\psi}[n+1] \right) \odot \frac{\partial G(\vec{z})}{\partial z} \Big|_{z=\vec{s}[n]} \quad (\text{B.45})$$

La ecuación B.42 y la ecuación B.44 muestran que las derivadas parciales totales de la función objetivo forman dos secuencias, que progresan en la dirección "hacia atrás" del índice, n . Estas secuencias representan las contrapartes duales de la secuencia generada al desenrollar la Ecuación B.34 en la dirección "hacia adelante" del índice, n .

Por lo tanto, así como la Ecuación B.34 requiere la inicialización de la señal de estado del segmento usando la Ecuación B.33, la secuencia formada por la derivada parcial total de la función objetivo con respecto a la señal de estado (comúnmente designada como "el gradiente de error") requiere que la Ecuación B.42 también debe inicializarse:

$$\vec{\psi}[n = K_m] = \vec{0} \quad (\text{B.46})$$

Aplicando la regla de la cadena a la Ecuación B.34 y usando la Ecuación B.42, se obtienen las expresiones para las derivadas de los parámetros del modelo:

$$\frac{\partial E}{\partial \Theta}[n] = \left\{ \frac{\partial E}{\partial W_r}[n], \frac{\partial E}{\partial W_x}[n], \frac{\partial E}{\partial \vec{\theta}_s} \right\} \quad (\text{B.47})$$

$$\frac{\partial E}{\partial W_r}[n] = \vec{\psi}[n] \vec{r}^T[n - 1] \quad (\text{B.48})$$

$$\frac{\partial E}{\partial W_x}[n] = \vec{\psi}[n] x_m^T[n] \quad (\text{B.49})$$

$$\frac{\partial E}{\partial \theta_s}[n] = \vec{\psi}[n] \quad (\text{B.50})$$

$$\frac{\partial E}{\partial \Theta}[n] = \sum_{n=0}^{K_m-1} \frac{\partial E}{\partial \vec{\Omega}[n]} \quad (\text{B.51})$$

Para una celda RNN, desenrollada para k_m pasos cubriendo segmentos de muestras de entrenamiento de largo K_m , el mismo conjunto de parámetros del modelo Θ , es compartido por todos los pasos. Esto se debe a que Θ es el parámetro del sistema RNN en su conjunto. En consecuencia, la derivada total de la función objetivo, E , con respecto a los parámetros del modelo, Θ , debe incluir las contribuciones de todos los pasos de la secuencia desenrollada.

Esto se captura en la Ecuación B.50, que ahora se puede usar como parte de la optimización de entrenamiento. Otra observación clave es que de acuerdo con la Ecuación B.47, Ecuación B.48 y Ecuación B.49, todas las cantidades esenciales para

actualizar los parámetros del sistema, Θ , durante el entrenamiento son directamente proporcionales a $\vec{\psi}[n]$.

Cuando el sistema RNN se entrena utilizando BPTT, la señal de gradiente de error fluye en la dirección inversa del índice, n , de la propia secuencia. Sea $\langle \vec{\psi}[k] \rangle_{0 \leq k \leq n}$ todos los términos de la secuencia, donde cada elemento, $\vec{\psi}[k]$, es el gradiente de E con respecto a la señal de estado, $\vec{s}[k]$, en el índice, k , para todo $k < n$, terminando en $\vec{\psi}[k=0]$. Entonces la Ecuación B.45 revela que $\vec{\psi}[n]$, el gradiente de E con respecto a la señal de estado, $\vec{s}[n]$, en algún valor del índice, n , sin importar cuán grande sea, puede influir en todo el conjunto, $\langle \vec{\psi}[k] \rangle_{0 \leq k \leq n}$. Por tanto, es de particular interés la fracción de $\vec{\psi}[n]$ que se retiene de la propagación inversa $\vec{\psi}[l]$, donde l es muy mayor a n . Este componente del gradiente de la función objetivo es responsable de ajustar los parámetros del modelo, Θ , de una manera que utiliza la información disponible en una muestra para reducir el costo del sistema que comete un error en una muestra distante. Si estos tipos de contribuciones a $\vec{\psi}[n] \rangle_{0 \leq n \leq k_m - 1}$ se comportan bien numéricamente, entonces los parámetros del modelo aprendidos mediante el procedimiento de optimización podrán incorporar las interacciones de largo alcance entre las muestras en la ventana RNN de manera efectiva durante la inferencia.

Expandiendo la recursividad en la Ecuación B.45 desde el paso con el índice, n , al paso con el índice, $l \leq k_m - 1$, donde $l \gg n$, da:

$$\frac{\partial \vec{\psi}[n]}{\partial \vec{\psi}[l]} = \prod_{k=n+1}^l W_r \odot \frac{dG(\vec{z})}{d\vec{z}}|_{z=\vec{s}[k]} \quad (\text{B.52})$$

De la ecuación B.52, la magnitud de la matriz jacobiana general, $\frac{\partial \vec{\psi}[n]}{\partial \vec{\psi}[l]}$, depende del producto de $l - n$ matrices jacobianas individuales, $\prod_{k=n+1}^l W_r \odot \frac{dG(\vec{z})}{d\vec{z}}|_{z=\vec{s}[k]}$ ¹¹.

A pesar de que el sistema RNN truncado y desenrollado es estable por diseño, en el caso de interacciones de largo alcance el tamaño de la ventana desenrollada, K_m , y la distancia entre las muestras de interés, $l - n$, son ambos grandes, el análisis de estabilidad es útil para estimar la magnitud de $\frac{\partial \vec{\psi}[n]}{\partial \vec{\psi}[l]}$ en la Ecuación B.52. Si todos los valores propios, μ_i i, de W_r satisfacen el requisito de estabilidad, $0 < \mu_i < 1$, entonces $\|W_r\| < 1$. Combinado con el hecho de que $\|\frac{dG(\vec{z})}{d\vec{z}}\| < 1$, esto produce:

$$\left\| \frac{\partial \vec{\psi}[n]}{\partial \vec{\psi}[l]} \right\| \sim \left(\|W_r\| \cdot \left\| \frac{dG(\vec{z})}{d\vec{z}} \right\| \right)^{l-n} \quad (\text{B.53})$$

¹¹Por convención, la multiplicación de elementos por un vector es equivalente a la multiplicación por una matriz diagonal, en la que los elementos del vector ocupan la diagonal principal.

$$\sim \|W_r\|^{l-n} \cdot \left\| \frac{dG(\vec{z})}{d\vec{z}} \right\|^{l-n} \approx 0 \quad (\text{B.54})$$

Por el contrario, si al menos un valor propio de W_r viola el requisito de estabilidad, el término $\|W_r\|^{l-n}$ crecerá exponencialmente. Esto puede llevar a dos posibles resultados para el sistema RNN en la Ecuación B.35.

En un escenario, como señal de estado, $\vec{s}[n]$, crece, los elementos de la señal de lectura, $\vec{r}[n]$, eventualmente se saturan en las regiones planas de la función de deformación. Dado que en el régimen de saturación, $\frac{dG(\vec{z})}{d\vec{z}} = \vec{0}$, el resultado es nuevamente $\left\| \frac{\partial \vec{\psi}[n]}{\partial \vec{\psi}[l]} \right\| \approx 0$.

En otro escenario, aunque raro, la señal de estado, $\vec{s}[n]$, está inicialmente sesgada en la región cuasi-lineal de la función de deformación, donde

$$\frac{dG(\vec{z})}{d\vec{z}} \neq \vec{0}$$

. Si la entrada, $xm[n]$, entonces guía al sistema para que permanezca en este modo durante una gran cantidad de pasos, $\left\| \frac{\partial \vec{\psi}[n]}{\partial \vec{\psi}[l]} \right\|$ crecerá, lo que podría resultar en un desbordamiento.

En consecuencia, el entrenamiento del sistema RNN estándar en ventanas que abarcan muchas muestras de datos utilizando el descenso del gradiente se ve obstaculizado por gradientes que desaparecen o explotan, independientemente de si el sistema es estable para señales grandes o no.

B.1.2. Long Short-Term Memory

Se pueden realizar varias modificaciones en el diseño de la celda para remediar esta situación. Como punto de partida, es útil separar el lado derecho de la Ecuación B.29 (la señal de estado actualizada de la celda en un paso con el índice, n) en dos partes.

$$\vec{s}[n] = \vec{\mathcal{F}}_s(\vec{s}[n-1]) + \vec{\mathcal{F}}_u(\vec{r}[n-1], \vec{x}[n]) \quad (\text{B.55})$$

$$\vec{r}[n] = G_d(\vec{s}[n]) \quad (\text{B.56})$$

$$\vec{\mathcal{F}}_s(\vec{s}[n-1]) = W_s \vec{s}[n-1] \quad (\text{B.57})$$

$$\vec{\mathcal{F}}_u(\vec{r}[n-1], \vec{x}[n]) = W_r \vec{r}[n-1] + W_x \vec{x}[n] + \vec{\theta}_s \quad (\text{B.58})$$

donde $G_d(\vec{z})$ es la tangente hiperbólica como antes ¹² . La primera parte, $\vec{\mathcal{F}}_s(\vec{s}[n-$

¹²A lo largo de este documento y en (Sherstinsky, 2020) , el subíndice “c” significa control”, mientras

1] , traslada la contribución de la señal de estado en el paso anterior. La segunda parte, $\vec{\mathcal{F}}_u(\vec{r}[n - 1], \vec{x}[n])$, representa la información de actualización, que consiste en la combinación de la señal de lectura del paso anterior y la señal de entrada (la fuerza impulsora externa) en la paso actual (más el vector de sesgo, $\vec{\theta}_s$) ¹³. Según la Ecuación B.55, la señal de estado combina ambas fuentes de información en proporciones iguales en cada paso. Estas proporciones se pueden hacer ajustables multiplicando las dos cantidades por las señales especiales de "puerta de olvido" ($\vec{g}_{cs}[n]$) y "puerta de entrada" ($\vec{g}_{cu}[n]$) :

$$\vec{s}[n] = \vec{g}_{cs}[n] \odot \vec{\mathcal{F}}_s(\vec{s}[n - 1]) + \vec{g}_{cu}[n] \odot \vec{\mathcal{F}}_u(\vec{r}[n - 1], \vec{x}[n]) \quad (\text{B.59})$$

$$\vec{0} \leq \vec{g}_{cs}[n], \vec{g}_{cu}[n] \leq \vec{1} \quad (\text{B.60})$$

Las señales de puerta, $\vec{g}_{cs}[n]$ y $\vec{g}_{cu}[n]$, en la Ecuación B.59 y la Ecuación B.60 proporcionan un mecanismo para ejercer un control detallado de los dos tipos de contribuciones a la señal de estado en cada paso. Específicamente, $\vec{g}_{cs}[n]$ permite controlar la cantidad de la señal de estado, retenida del paso anterior, y $\vec{g}_{cu}[n]$ regula la cantidad de la señal de actualización que se inyectará en la señal de estado en el paso actual.

W_s en Ecuación B.57 es una matriz diagonal con fracciones positivas, $\frac{1}{a_{ii}}$, en su diagonal principal. Por lo tanto, dado que los elementos de $\vec{g}_{cs}[n]$ también son fracciones, si se establece:

$$W_s = I \quad (\text{B.61})$$

en $\vec{g}_{cs}[n] W_s$ es aceptable siempre que las funciones de puerta sean parametrizables

que el subíndice "d" significa "datos"

¹³En sistemas discretos, el concepto de tiempo solo tiene importancia histórica. La terminología adecuada sería utilizar la palabra .^adyacente cuando se refiere a cantidades en los pasos vecinos. Aquí, los términos .^anterior" , .^aactual" "siguiente" se utilizan a veces sólo por motivos de conveniencia. Los sistemas RNN se pueden desenrollar fácilmente en la dirección opuesta, en cuyo caso se niegan todos los índices y se invierte el significado de .^anterior" "siguiente". De hecho, se ha logrado un rendimiento mejorado con el procesamiento bidireccional en una variedad de aplicaciones [15, 17, 32, 55, 58, 78]. Además, si los datos de entrada se preparan con anticipación y se ponen a disposición del sistema en su totalidad, entonces la restricción de causalidad se puede relajar por completo. Esto puede ser factible en aplicaciones, donde se recopila todo el conjunto de datos de entrenamiento o una colección de segmentos de datos de entrenamiento independientes antes de que comience el procesamiento. El procesamiento no causal (es decir, una técnica caracterizada por aprovechar los datos de entrada "del futuro") puede ser ventajoso para detectar la presencia de contexto.^{entre} las muestras de datos. Utilizar la información en los pasos "futuros" como parte del contexto para tomar decisiones en el paso .^actual" suele ser beneficioso para analizar audio, voz y texto.

y sus parámetros se aprendan durante el entrenamiento. Bajo estas condiciones, la Ecuación B.57 se puede simplificar a:

$$\vec{\mathcal{F}}_s(\vec{s}[n-1]) = \vec{s}[n-1] \quad (\text{B.62})$$

de modo que la Ecuación B.59 se convierte en:

$$\begin{aligned} \vec{s}[n] &= \vec{g}_{cs}[n] \odot \mathcal{F}_s(\vec{s}[n-1]) + \vec{g}_{cu}[n] \odot \mathcal{F}_u(\vec{r}[n-1], \vec{x}[n]) \\ &= \vec{g}_{cs}[n] \odot \vec{s}[n-1] + \vec{g}_{cu}[n] \odot \mathcal{F}_u(\vec{r}[n-1], \vec{x}[n]) \end{aligned} \quad (\text{B.63})$$

Mientras que el término de actualización, $\vec{\mathcal{F}}_u(\vec{r}[n-1], \vec{x}[n])$, como un todo ahora está controlado por $\vec{g}_{cu}[n]$, la composición interna de $\vec{\mathcal{F}}_u(\vec{r}[n-1], \vec{x}[n])$ debe ser examinado. Según la Ecuación B.58, la señal de lectura del paso anterior y la señal de entrada en el paso actual constituyen la señal candidata de actualización en cada paso con índice n, contribuyendo ambos términos en proporciones iguales. El problema de utilizar siempre $W_r \vec{r}[n-1]$ en su totalidad es que cuando $\vec{g}_{cu}[n] \sim 1$, $\vec{s}[n-1]$ y $\vec{s}[n]$ se conectan a través de W_r y la función de deformacion.

Con base en la Ecuación B.52, este vínculo restringe el gradiente de la función objetivo con respecto a la señal de estado, lo que predispone al sistema al problema de desaparición de gradientes. Para acelerar esta ruta de retroalimentación, la señal de lectura, $\vec{r}[n]$, será distribuida por otra señal, $\vec{g}_{cr}[n]$ llamada ("puerta de salida"):

$$\vec{v}[n] = \vec{g}_{cr}[n] \odot \vec{r}[n] \quad (\text{B.64})$$

$$0 \leq \vec{g}_{cr}[n] \leq \vec{1} \quad (\text{B.65})$$

El control de la puerta, $\vec{g}_{cr}[n]$, determina la cantidad fraccional de la señal de lectura que se convierte en la señal de valor observable de la celda en el paso con índice n. Por lo tanto, usar $\vec{v}[n-1]$ en lugar de $\vec{r}[n-1]$ en la Ecuación B.58 lo transforma en:

$$\vec{\mathcal{F}}_u(\vec{v}[n-1], \vec{x}[n]) = W_r \vec{v}[n-1] + W_x \vec{x}[n] + \vec{\theta}_s \quad (\text{B.66})$$

El diagrama de la celda RNN, expandido donde se acomoda la puerta de control de lectura, introducido en la Ecuacion B.64, y la relación de recurrencia modificada, empleada en la Ecuación B.66, aparece en ilustracion 2.4.

Anexo C

Transformadores

C.1. Características de Transformadores

Un transformador puede realizar casi cualquier tarea de NLP. Se puede utilizar para modelado, traducción o clasificación de idiomas según sea necesario, y lo hace rápidamente al eliminar la naturaleza secuencial del problema. Todo dependerá de la capa de salida final de la red, pero la estructura básica de Transformer seguirá siendo la misma para cualquier tarea.

Un transformer se compone esencialmente de una pila de capas de codificador y otra de decodificador, representados en la ilustración C.1 como bloques grises izquierdo y derecho respectivamente . Aquí, el codificador mapea una secuencia de entrada de representaciones de símbolos (x_1, \dots, x_n) a una secuencia de representaciones continuas $z = (z_1, \dots, z_n)$. Dado z , el decodificador genera una secuencia de salida (y_1, \dots, y_m) de símbolos un elemento a la vez. En cada paso, el modelo es auto-regresivo (Graves, 2014), consumiendo los símbolos generados previamente como entrada adicional al generar el siguiente.

La unidad fundamental de esta arquitectura es la atención. Existen muchos mecanismos de atención, pero en un transformer se calcula para cada palabra tres componentes.

- Una clave o key : El "perfil" de la palabra, por ejemplo el perfil de "bicicleta" puede incluir "Soy un sustantivo de género neutro".
- Una consulta o query: Responde a qué busca la palabra cuando rastrea perfiles (un pronombre como "eso" podría decir coincidir con sustantivos de género neutro")

- Un valor o value: Otra información sobre lo que significa la palabra, que podría no ser tan relevante para este proceso de coincidencia (por ejemplo, todo lo demás sobre lo que significa la palabra "bicicleta")

Para cada palabra se utilizan las claves y las consultas para averiguar cuánto coincide la palabra con ella misma y cuánto coincide entre sí. Luego, se suman los valores ponderados por las "puntuaciones de coincidencia". Al final, es posible que obtengas algo que combine gran parte del valor original de la palabra con un poco de los valores de otras palabras, lo que representa algo como "Sigo siendo un pronombre, pero también, sustituyó a este sustantivo y me refiero a lo mismo que él".

Dado que las palabras se pueden relacionar de muchas formas diferentes, es un poco restrictivo tener solo un conjunto (clave / consulta / valor) por palabra. Por ende es posible agregar tantos conjuntos como uno quiera por palabra. Esto se denomina atención de "múltiples encabezados" (bloque naranja de la ilustración C.1), donde el número de claves / consultas / valores por palabra es el "número de cabezas de atención".

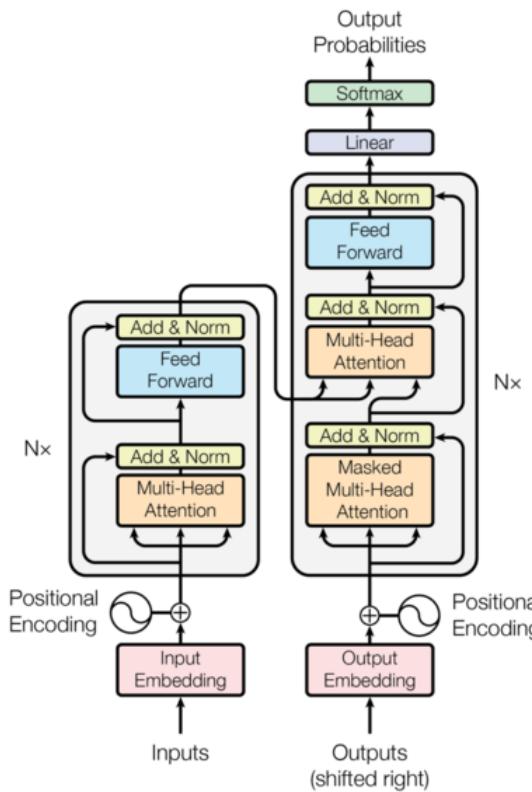


Ilustración C.1: Diagrama de transformador ilustrada en (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser & Polosukhin, 2017)

C.2. Arquitectura de codificador

Recordar que las capas del codificador se apilan una encima de la otra. Por lo tanto, es necesario tener una salida y una entrada de iguales dimensiones . Cada entrada y salida de un codificador dentro de la pila requiere una forma matricial de dimensiones $S \times D$, donde S es la longitud de la oración fuente y D es la dimensión de la incrustación (embedding) ¹.Cada codificador tiene en su interior dos capas principales, una capa de auto-atención de múltiples encabezados y una red feed-forward completamente conectada en función de la posición.

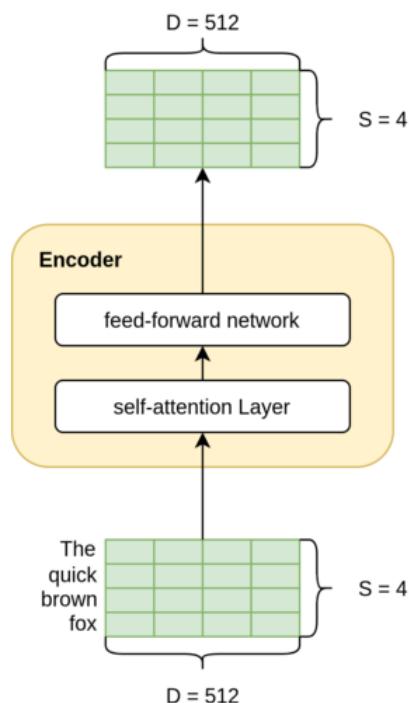


Ilustración C.2: Ejemplo ilustrativo de la estructura del encoder con $D=512$ y $S=4$ que representa la duración máxima de la sentencia según el lote

¹Una incrustación o embedding es un espacio de dimensiones relativamente bajas en el que se pueden traducir vectores de dimensiones altas. Las incrustaciones facilitan el aprendizaje automático en entradas grandes, como vectores dispersos que representan palabras. Idealmente, una incrustación captura parte de la semántica de la entrada colocando entradas semánticamente similares juntas en el espacio de incrustación. Una incrustación se puede aprender y reutilizar en todos los modelos.

La capa de auto-atención se inicializa con 3 matrices de peso, que representan las componentes bases (Clave (W_k), Consulta (W_q) y Valor (W_v), cuyas dimensiones son $D \times D^2$. Los pesos de estas matrices se entrenaron junto con el modelo.

Primero se crean las matrices Q , K y V multiplicando la entrada con la correspondiente matriz de Consulta, Clave y Valor. Luego se realiza un segundo cálculo multiplicando la matriz Q con K transpuesta obteniendo una matriz de dimensiones $S \times S$, la cual se divide por \sqrt{d} y se entrega como entrada a una función softmax para obtener la suma de sus filas igual a 1. Esta matriz resultante representa las contribuciones de cada palabra con respecto a las otras, entregando valores más altos en aquellas palabras que estén más relacionadas. Finalmente multiplicamos esta matriz por la matriz Valor, de dimensiones $S \times D$, y nos devuelve otra matriz de forma $S \times D$. Esta operación permite retornar el vector de la palabra embebida con ponderaciones apropiadas a las contribuciones con las otras palabras dando así la atención dentro de la oración.

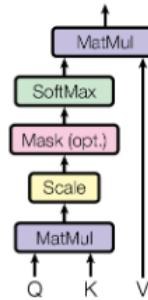


Ilustración C.3: Atención de productos escalados

Al ser una capa de múltiples encabezados, la entrada X atraviesa muchas capas de auto-atención en paralelo, cada una de las cuales da una matriz z de dimensiones ($S \times D$). Se concatenan estas matrices, siendo h el número de capas de auto-atención en paralelo, y nuevamente se aplica una capa lineal de salida final, W_0 , de tamaño $D \times D$. Finalmente se obtiene una matriz de tamaño $S \times D$, cuya dimensionalidad permite ingresar al siguiente codificador.

La red Feed-forward completamente conectada en función de la posición es una combinación de varias capas lineales y dropout. Esta red se aplica a cada posición de la matriz, proveniente de la capa de auto-atención, en paralelo, donde cada posición se puede considerar como una palabra.

²d se toma como 64 en (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser & Polosukhin, 2017)

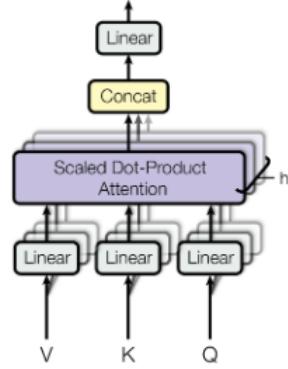


Ilustración C.4: La atención de múltiples encabezados consta de varias capas de atención, ilustración C.3, que se ejecutan en paralelo

Dado que este modelo no contiene recurrencia ni convolución, para que el modelo haga uso del orden de la secuencia, es necesario inyectar alguna información sobre la posición relativa o absoluta de los tokens en la secuencia. Con este fin, se agregan los "codificaciones posicionales" a las incrustaciones de entrada, se puede ver en la ilustración C.1 como símbolos sinusoidales ya que los autores utilizan funciones de seno y coseno para crear incrustaciones (embedding) posicionales para diferentes posiciones. Finalmente está el bloque que adiciona la matriz de entrada al bloque con la salida para luego normalizar. Esto sirve básicamente para ayudar al flujo de información en arquitecturas de un gran número de capas.

C.3. Arquitectura del decodificador

Como se puede ver en la ilustración C.1 el decodificador presenta bloques similares al codificador y dos entradas, Una proveniente desde la salida del codificador y la otra desde la salida desfasada del mismo decodificador. Esto nos indica que un transformador se debería entender como un modelo de lenguaje condicional, es decir que predice la siguiente palabra dada una palabra de entrada y una oración sobre la cual condicionar o basar su predicción. Este tipo de modelos son inherentemente secuenciales, ya que el proceso de aprendizaje parte con un primer token (js_t) y el modelo predice la primera palabra condicionada a la oración proveniente del codificador cambiando los pesos en función de si la predicción es correcta o incorrecta. Luego se le entrega el token de inicio y la primera palabra y el modelo predice la segunda palabra para luego cambiar los pesos nuevamente y así sucesivamente hasta llegar al token final. Si bien el decodificador del transformador aprende de esta manera, no lo

hace de forma secuencial. Este utiliza enmascaramiento para hacer este cálculo y, por lo tanto, toma toda la oración de salida (aunque se desplaza a la derecha agregando un token `js` al principio) durante el entrenamiento.

Para realizar este enmascaramiento se utiliza, en forma adicional, un bloque de atención de múltiples encabezados enmascarados. Esto significa que se enmascara la salida desplazada (que es la entrada al decodificador) de una manera que la red nunca podrá ver las palabras subsiguientes, ya que de lo contrario, podría copiar fácilmente esa palabra mientras se entrena.

Como se señaló anteriormente, en la capa de atención se multiplica la consulta (Q) y las claves (K) para luego dividirlo por \sqrt{d} antes de pasar por la función softmax. Sin embargo, en una capa de atención enmascarada, se agrega a la matriz resultante antes del softmax (que tendrá dimensiones ($T \times T$)) una matriz de enmascaramiento. De esta forma elimina las componentes de las palabras siguientes, es decir todas las posiciones posteriores a la palabra tiene valor 0 luego de pasar por la función softmax. Ahora, si multiplicamos esta matriz con la matriz de valores (V), el vector correspondiente a la posición de la palabra en el vector Z que pasa por el decodificador no contendrá ninguna información de las siguientes palabras. De esta forma es como el transformador toma toda la oración de salida desplazada de una vez y no aprende de manera secuencial.

Por otro lado en la capa de atención de múltiples cabezales sin enmascarar, el vector de consulta (Q) se crea a partir de los datos que fluyen desde la capa de atención enmascarada, mientras que los vectores de Clave (K) y Valor (V) provienen de la salida del codificador. Como la salida que viene de abajo ya está enmascarada, esto permite que cada posición en el decodificador preste atención a todas las posiciones en el vector Valor. Por ende, para cada posición de palabra que se genere, el decodificador tiene acceso a la oración completa.

Anexo D

Resultados

D.1. Experimento 1: Graficos

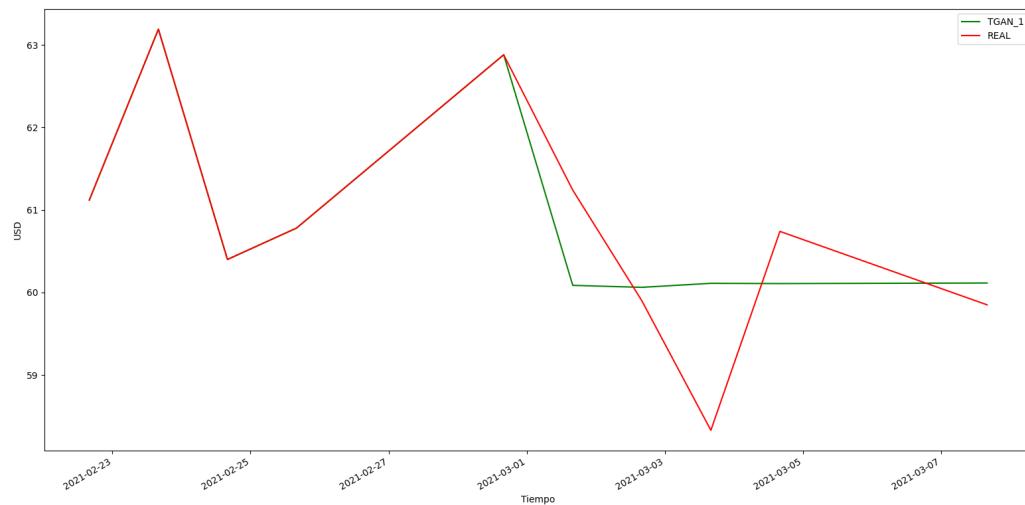


Ilustración D.1: Experimento 1. Gráfico Comparativo en TimeGAN original ('INTC').
Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]

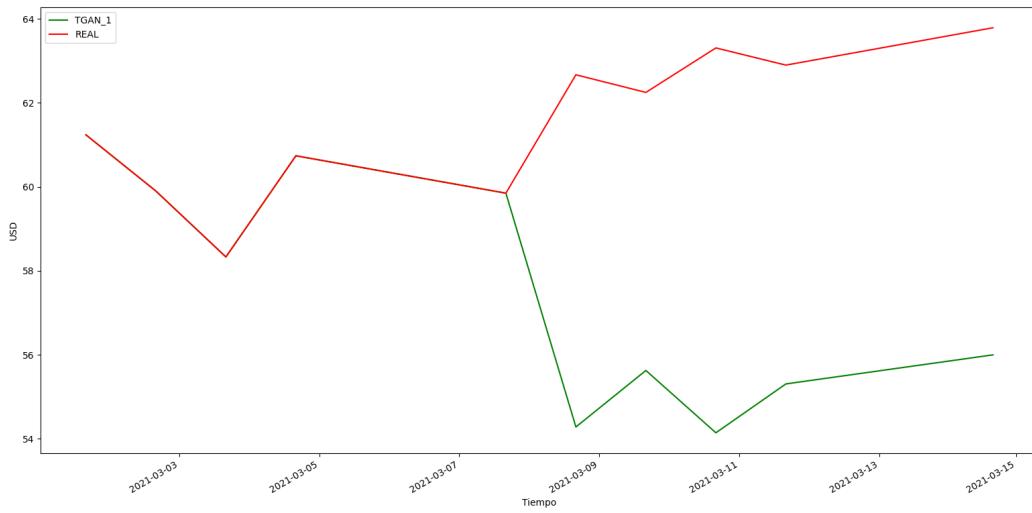


Ilustración D.2: Experimento 1. Gráfico Comparativo en TimeGAN original ('INTC').
 Ventana evaluada : [2021-03- 02/2021-03- 08]. Ventana predictiva : [2021-03-9/2021-03-15]

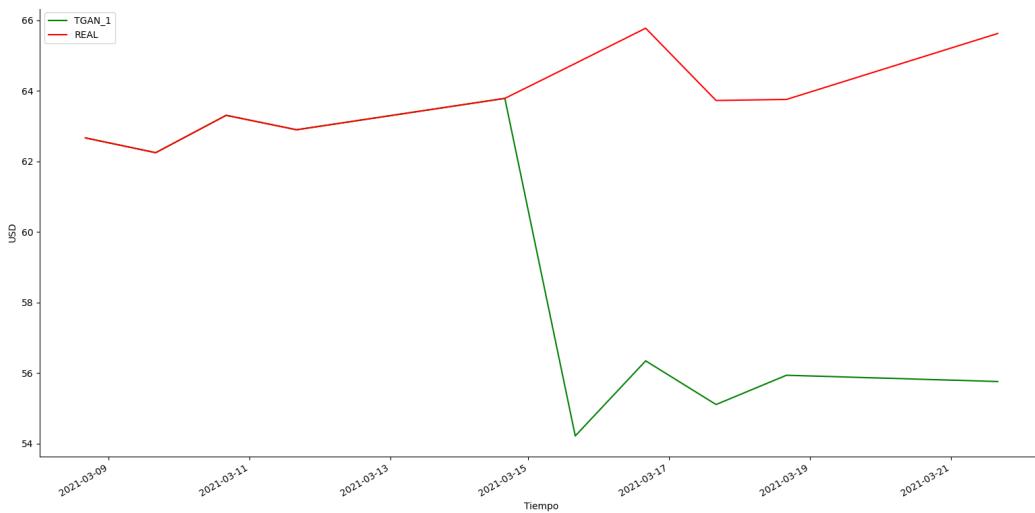


Ilustración D.3: Experimento 1. Gráfico Comparativo en TimeGAN original ('INTC').
 Ventana de Datos : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]

D.2. Experimento 3: Graficos

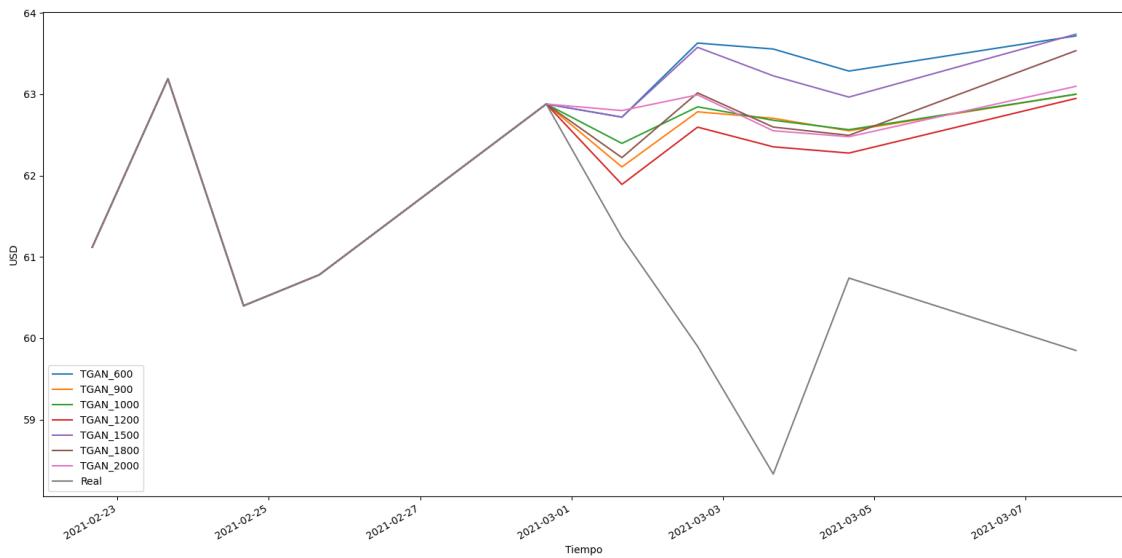


Ilustración D.4: Experimento 3. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]

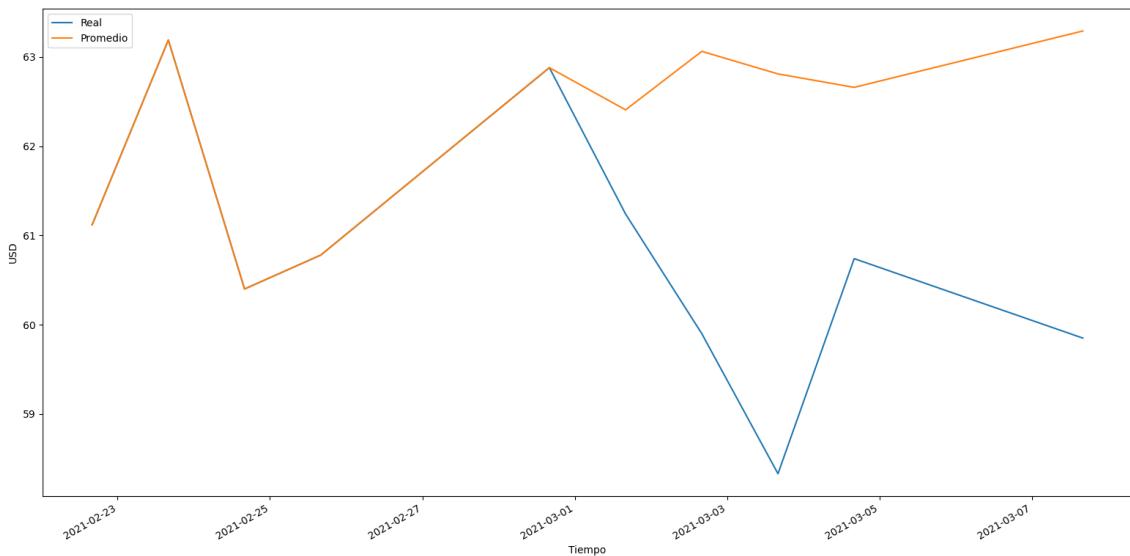


Ilustración D.5: Experimento 3. Gráficos Comparativos Promedio ('INTC'). Ventana evaluada : [2021-02- 23/2021-03- 01]. Ventana predictiva : [2021-03-02/2021-03-08]

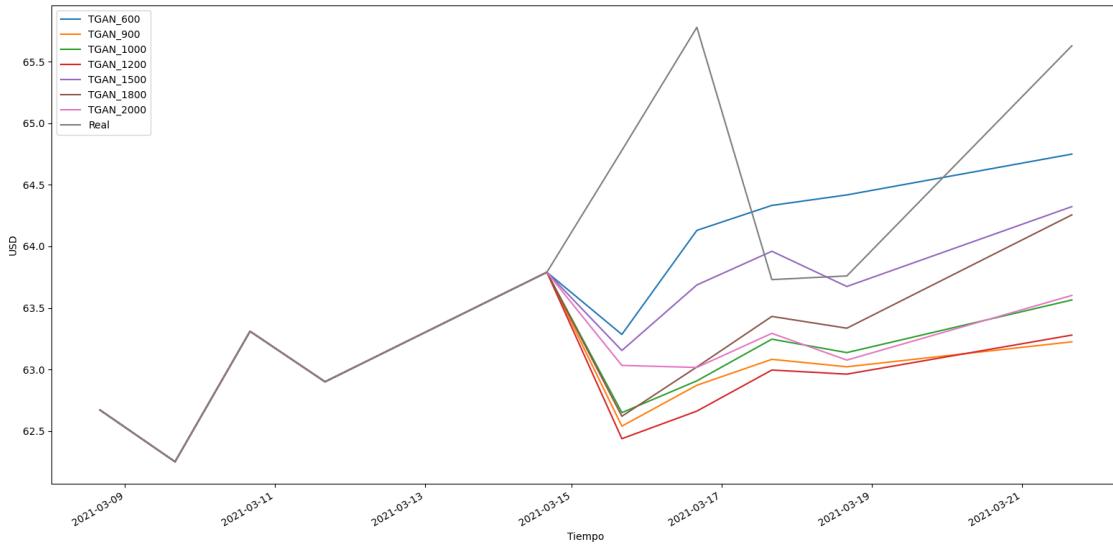


Ilustración D.6: Experimento 3. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]

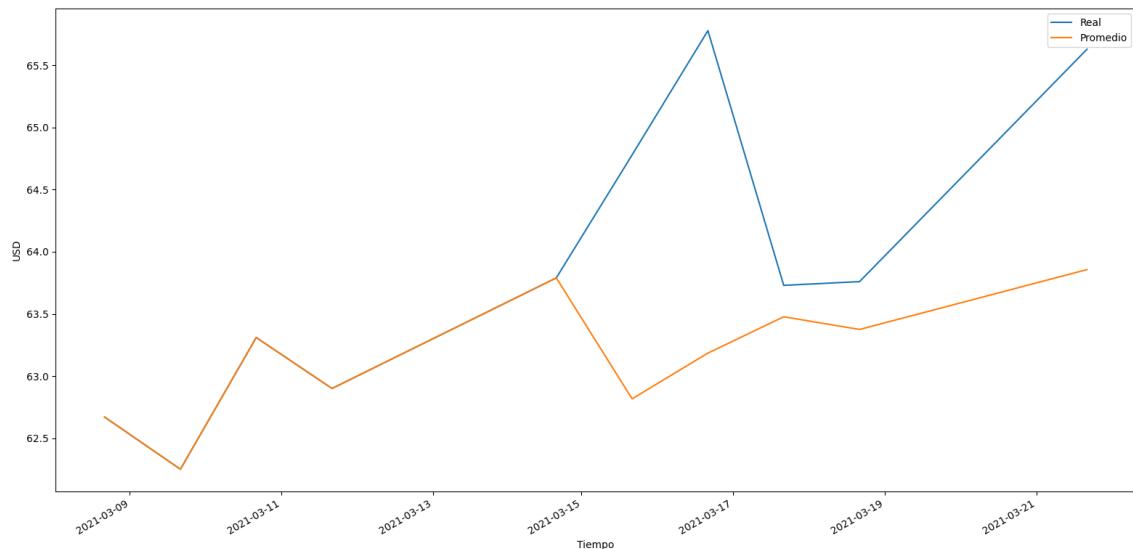


Ilustración D.7: Experimento 3. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [2021-03- 09/2021-03- 15]. Ventana predictiva : [2021-03-16/2021-03-22]

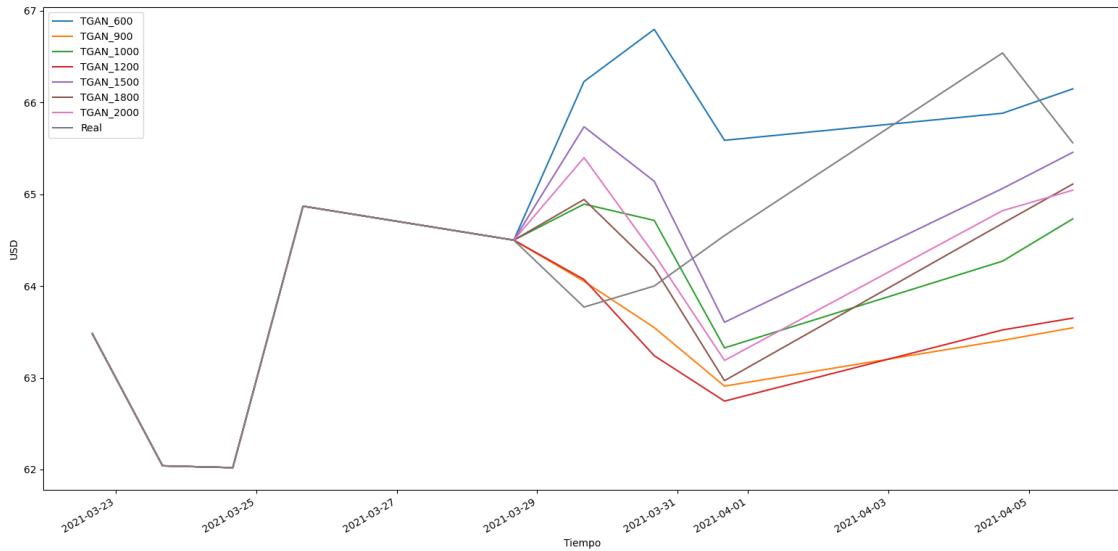


Ilustración D.8: Experimento 3. Gráfico Comparativo Completo ('INTC'). Ventana evaluada : [2021-03- 23/2021-03- 29]. Ventana predictiva : [2021-03-30/2021-04-05]

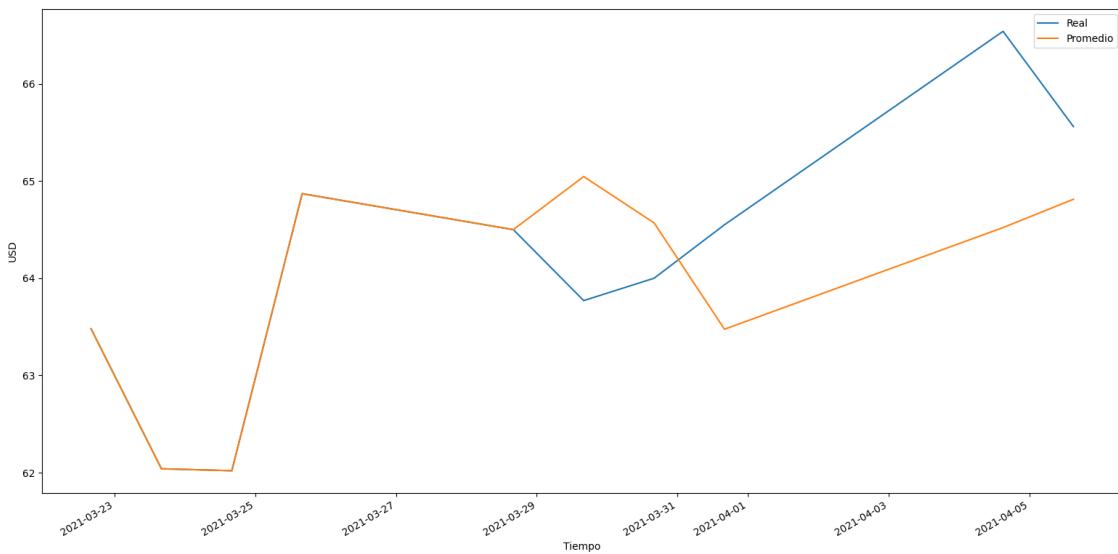


Ilustración D.9: Experimento 3. Gráfico Comparativo Promedio ('INTC'). Ventana evaluada : [[2021-03- 23/2021-03- 29]. Ventana predictiva : [2021-03-30/2021-04-05]]

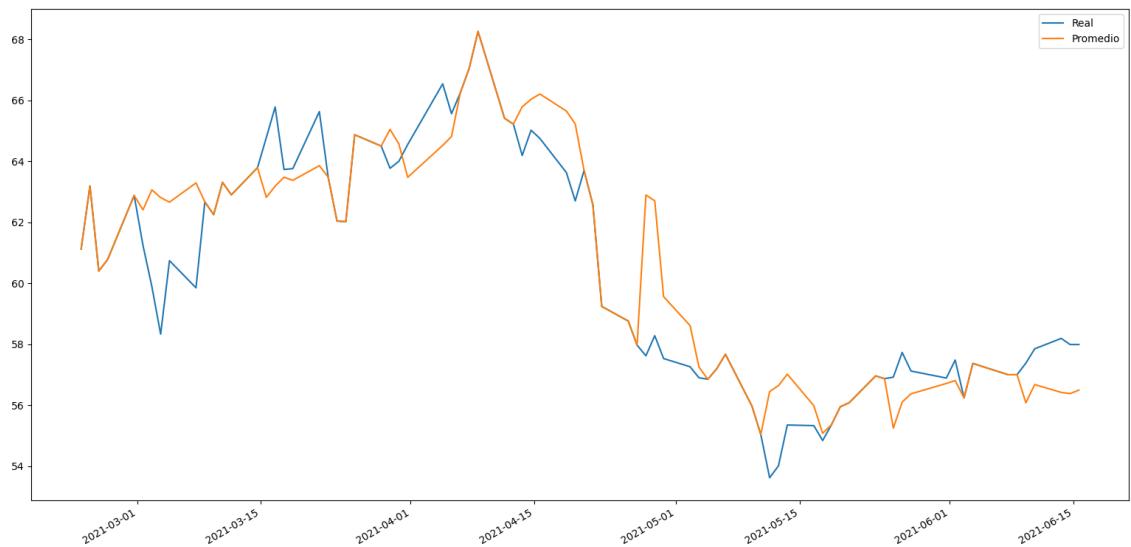


Ilustración D.10: Experimento 3. Gráfico Comparativo Promedio ('INTC') con todo el conjunto de datos de evaluación . Conjunto de evaluación : [2021-02- 23/2021-06-15]