**Pronóstico de Series de Tiempo de Tráfico web. Estudio de Caso Universidad Nacional de Misiones**

POR: Enrique Gauto Sand

Propuesta de Tesis presentada a la Facultad de Ciencias Exactas, Químicas y Naturales de la Universidad Nacional de Misiones para optar al grado académico / título profesional de Licenciado en Sistemas de Información de la carrera de Licenciatura en Sistemas de Información

Directora: Alice Rambo

Prof. Tit. de la Cátedra “Trabajo Final”: Dr. Kuna, Horacio Daniel

Índice

[Capítulo 1 1](#_Toc164192957)

[Introducción 1](#_Toc164192958)

[1.1. Motivación 2](#_Toc164192959)

[1.2. Objetivos 2](#_Toc164192960)

[1.2.1. Objetivo General: 2](#_Toc164192961)

[1.2.2. Objetivos Específicos: 2](#_Toc164192962)

[1.3. Estructura del documento 2](#_Toc164192963)

[Capítulo 2 3](#_Toc164192964)

[Marco Teórico 3](#_Toc164192965)

[2.1. Tráfico web 4](#_Toc164192966)

[2.2. Series de tiempo 4](#_Toc164192967)

[2.1. Pronóstico de series de tiempo de tráfico web 4](#_Toc164192968)

[2.2. Evaluación de desempeño de los modelos predictivos 6](#_Toc164192969)

[Capítulo 3 7](#_Toc164192970)

[Descripción del problema 7](#_Toc164192971)

[3.1 El problema 8](#_Toc164192972)

[3.2 Estructura Organizacional 8](#_Toc164192973)

[Organigrama 8](#_Toc164192974)

[Capítulo 4 10](#_Toc164192975)

[Solución Propuesta 10](#_Toc164192976)

[4.1 Materiales y Métodos 11](#_Toc164192977)

[4.2 Herramientas 12](#_Toc164192978)

[4.2.1 Python 12](#_Toc164192979)

[4.2.2 Google Drive 12](#_Toc164192980)

[4.2.3 Universal Analytics (UA) 13](#_Toc164192981)

[4.2.4 Google Analytics 4 (GA4) 13](#_Toc164192982)

[4.2.5 Looker Studio 13](#_Toc164192983)

[4.2.6 Microsoft Word 13](#_Toc164192984)

[4.2.7 Microsoft Excel 13](#_Toc164192985)

[4.2.8 Numpy 13](#_Toc164192986)

[4.2.9 Tensorflow 13](#_Toc164192987)

[4.2.10 Keras 14](#_Toc164192988)

[4.2.11 Pandas 14](#_Toc164192989)

[4.2.12 Matplotlib 14](#_Toc164192990)

[4.2.13 Google Colaboratory 14](#_Toc164192991)

[4.2.14 Git 14](#_Toc164192992)

[4.2.15 Hyperparametros (*Hyperparameters*) 14](#_Toc164192993)

[4.2.16 Época (*Epoch)* 14](#_Toc164192994)

[4.2.17 Trial 15](#_Toc164192995)

[4.2.18 Unidad Recurrente con Puerta – GRU (*Gate Recurrent Unit*) 15](#_Toc164192996)

[4.2.19 Arquitectura Codificador-Decodificador (*Encoder-Decoder*) 17](#_Toc164192997)

[4.2.20 Algoritmo Optimización Bayesiana (*Bayesian Otimization*) 17](#_Toc164192998)

[4.2.21 Algoritmo Hyperband 17](#_Toc164192999)

[4.2.22 Algoritmo Búsqueda Aleatoria (*Random Search*) 17](#_Toc164193000)

[4.2.23 Detención Temprana (*Early Stopping)* 18](#_Toc164193001)

[4.2.24 Github 18](#_Toc164193002)

[4.3 Comprensión de los datos 19](#_Toc164193003)

[4.3.1 Recolección de datos iniciales 19](#_Toc164193004)

[4.3.2 Preparación de los datos 20](#_Toc164193005)

[5 Modelos 1](#_Toc164193006)

[3.5 Selección de Modelos 1](#_Toc164193007)

[Desarrollo 2](#_Toc164193008)

[Conversión de Modelos de UA a GA4 de Tum Transmedia 10](#_Toc164193009)

[Conversión de Modelos de UA a GA4 de Editorial Universitaria 10](#_Toc164193010)

[Capítulo 5 14](#_Toc164193011)

[Conclusión 14](#_Toc164193012)

[5.1 Conclusiones 15](#_Toc164193013)

[5.2 Futuras líneas de investigación 15](#_Toc164193014)

[Anexo 17](#_Toc164193015)

[Bibliografía 19](#_Toc164193016)

# Capítulo 1

# Introducción

## Motivación

## Objetivos

### Objetivo General:

Pronosticar series de tiempo de tráfico web y medir el desempeño de modelos aplicados sobre datos extraídos de las páginas institucionales de la U.Na.M.

### Objetivos Específicos:

-Relevar y analizar distintos modelos de pronóstico de series de tiempo de tráfico web.

-Determinar la adecuación de cada uno de los modelos propuestos de la bibliografía a la problemática.

-Definir indicadores para medir el desempeño de los modelos.

-Generar los modelos a utilizar.

-Realizar la evaluación del desempeño de los modelos.

-Validar los modelos.

## Estructura del documento

# Capítulo 2

# Marco Teórico

En este capítulo se presenta el marco teórico del trabajo realizado, comenzando con la definición de tráfico web y series de tiempo, luego con variados ejemplos de la literatura de distintas tecnologías que realizan el pronóstico de series de tiempo de tráfico web finalizando con las formas de medir el desempeño de los modelos.

## Tráfico web

El trafico web es generado por los usuarios de una página web, el trafico web son los datos que se envían y se reciben correspondientes a los usuarios que visitan la página web[1].

## Series de tiempo

Las series de tiempo son un conjunto de valores medidos en orden secuencial en el tiempo[2]; cuando se miden los datos para generar una serie de tiempo, generalmente se toman los valores con la misma separación en tiempo entre cada valor[3].

## Pronóstico de series de tiempo de tráfico web

El pronóstico de series de tiempo se basa en las observaciones pasadas de la serie de tiempo a pronosticar y otras entradas, siendo el proceso de predecir valores futuros de una serie de tiempo [3].

Entre las formas del pronóstico de series de tiempo de tráfico web los autores en[4] proponen redes Modelo Generativo Adversario - GAN(*Generative Adversarial model*)[4] con reders de Memoria a Corto y Largo Plazo – LSTM (*Long Short Term Memory*)[5] y un perceptrón multicapa - MLP (*Multilayer Perceptron*)[6], donde LSTM y GAN actuarían como generador y el MLP como discriminador, para generar series de tiempo dado el conjunto de datos real, finalmente se realizaría el pronóstico con la librería *Prophet*[7] comparando la combinación de tecnologías anteriores con métodos estadísticos, finalmente llegando a la conclusión de que los autores no obtuvieron una diferencia notable[4].

Entre las formas del pronóstico de series de tiempo de tráfico web que existen, los autores en [8] proponen la técnica de redes neuronales LSTM[5] con entrenamiento asíncrono distribuido, cuya métrica de desempeño que se utilizó según el autor en[8] es el Error Medio Absoluto - MAE (*mean absolute error*)[9] y la función de pérdida de Huber[10] para probar la precisión del modelo, han logrado un buen grado de asertividad con 200 épocas, en el documento científico[8] mencionan que obtuvieron una MAE en promedio menor que 30, lo que consideran un buen resultado[8].

Hay otro artículo[1] en el cual se menciona el uso de Redes Neuronales Recurrentes -RNN (*Recurrent Neural Networks*) seq2seq[1] con la ayuda de la arquitectura Codificador –Decodificador (*encoder/decoder)*[11], el *encoder* es cuDNN-GRU

Es decir una Unidad Recurrente con Puerta (*Gate Recurrent Unit*)[12] con backend de CUDA, ya que realiza la tarea con mayor velocidad en comparación con los tensores regulares, el *decoder* es TensorFlow GRUBlockCell[12], en este documento científico[1] menciona que realizaron algunos cambios al modelo ganador de *Kaggle*[12] como agregar la mediana de 7,30,90 y 180 días, usaron para medir el desempeño del modelo la métrica Error Absoluto Medio Porcentual Simétrico -SMAPE (*Symmetric Mean Absolute Percentage Error*)[13] donde comentan los autores que obtuvieron un SMAPE de 0.349[1].

En otro artículo[14], proponen utilizar un enfoque híbrido de optimización de enjambre de partículas (*particle swarm optimization)* e inspiración cuántica (*quantum-inspired)* resultando en QPSO[14] y redes Exogenas Autorregresivas No Lineales – NARX(*Nonlinear autoregressive exogenous*)[15] resultando en NARX -QPSO[14] en dicho artículo se menciona que el modelo NARX –QPSO tiene un rendimiento sobresaliente en los resultados comparados con otros modelos[14].

En otro artículo[16] propone un diseño de un sistema Aprendizaje Automático Automatizado - *Automated Machine Learning* (Auto ML) [16], una arquitectura neuronal nueva denominada Auto-PyTorch-TS[16] donde lo comparan con otros modelos y demuestran que su modelo tiene un mejor rendimiento[16].

También se propone en el artículo[17] el uso de un modelo combinado

*Prophet* y Maquina de Empuje de Gradiente de Luz *-* LGBM(*light gradient boosting machine*)[17] donde se menciona que tiene mejores resultados comparado con modelos individuales[17].

Otra tecnología son las redes bidireccionales LSTM (BI-LSTM)[2] las cuales en este artículo[2] fueron probadas en el conjunto de datos *M3-Competition*[18]el cual se usa para probar modelos de pronóstico de series de tiempo dado que posee distintas categorías, en el artículo[2] mencionan que probaron las redes BI-LSTM sobre el conjunto de datos M3 en el periodo trimestral comparándolo con el modelo Media móvil integrada autorregresiva ARIMA (*Auto-Regressive Integrated Moving Average*)[3] y otros, donde BI-LSTM tuvo un mejor rendimiento.

## Evaluación de desempeño de los modelos predictivos

El desempeño es la forma de evaluar el modelo, se utiliza para comparar los valores predichos del modelo entrenado, con los valores observados[19]

La Distancia media cuadrática mínima - RMSE(*Root Mean Square Errors*): el error cuadrático medio es una medida de las diferencias entre los valores predichos por un modelo y los valores observados.

(1)

Donde es el valor predicho y es el valor real[9]

El error medio absoluto - MAE(*mean absolute error*)[9] se calcula de la siguiente forma:

(2)

Donde es el valor predicho y es el valor real[9],

La métrica Error Absoluto Medio Porcentual Simétrico - SMAPE (*Symmetric Mean Absolute Percentage Error*)[13] se calcula de la siguiente forma:

(3)

Siendo el valor pronosticado, es el valor observado y n es el tamaño de la muestra.

Si bien existe otra alterativa a la fórmula de SMAPE donde se toman los valores absolutos de en el denominador, en este caso se optó por la fórmula original, ya que entre los datos no tenemos valores negativos de vistas.

# Capítulo 3

# Descripción del problema

En este capítulo se describe la problemática a abarcar como así también la estructura organizacional.

## **El problema**

El presente trabajo pretende hacer análisis de tráfico web para poder estimar y predecir futuras demandas en las páginas institucionales de la Universidad Nacional de Misiones – U.Na.M. Para lo cual se consideraran indicadores como volumen de tráfico, cantidad de conexiones, cantidad de visitantes, velocidad de carga de las paginas, páginas vistas, promedio de páginas por vista, porcentaje de rebote, entre otros indicadores según la disponibilidad de los mismos. Siendo provisto por parte del personal técnico referente institucional acceso a las herramientas de captura de datos estadístico de los siguientes enlaces:

* Sitio Institucional de la U.Na.M. disponible en <https://unam.edu.ar/>
* Sitio de la editorial Universitaria disponible en <https://editorial.unam.edu.ar/>
* Portal de acceso al contenido generado por la radio y la televisión de la Universidad disponible en <https://transmedia.unam.edu.ar/>

De los sitios mencionados anteriormente se extraerán los datos para utilizar en este proyecto con la finalidad de realizar el pronóstico de series de tiempo de tráfico web. Inicialmente como herramienta de acceso a los datos se dispone *Google analytics* con permisos cedidos por los administradores.

## Estructura Organizacional

Con el fin de e1 entender el funcionamiento del Departamento de Gestión de Recursos de Redes y Comunicaciones se procede a describir su estructura organizacional, como así también los roles de los trabajadores.

### Organigrama



Figura 1: Organigrama del Departamento de Gestión de Recursos de Redes y Comunicaciones

El Departamento de Gestión de Recursos de Redes y Comunicaciones, gestiona la infraestructura, además pone a disposición las herramientas para la puesta en servicio de las páginas web y servicios añadidos. Pero no controla los contenidos, diseño o función de los sitios.

* **Jefe de departamento**: Realiza las tareas propias del jefe de departamento, también se encarga de mantener las plataformas de los sitios web, por ejemplo Moodle para Aulas Virtuales.
* **Técnico 1:** Se encarga principalmente de la gestión y mantenimiento de los servicios de voz sobre protocolo de Internet - VoIP, redes WiFi, gestión de las Redes de Área Local Virtual - VLAN (Virtual Local Area Network) y redes de la unidad central de Rectorado, gestión de usuarios en la plataforma de *Google Workspace for education*, entre otros.
* **Técnico 2 y Técnico 3:** Estos dos técnicos se encargan de la gestión  y ruteo de las redes de la universidad y con los proveedores de servicio, gestión y mantenimiento de los Servidores físicos y virtuales del Centro de Procesamiento de Datos - CPD, plataformas de respaldo de datos, sistemas de monitoreo de las plataformas y servicios esenciales de la infraestructura de la universidad como Sistema de Nombres de Dominio – DNS (agregar), Protocolo de Puerta de Enlace Fronteriza – BGP, Camino Más Corto Primero – OSPF (*Open Shortest Path First*), Red Privada Virtual – VPN (Virtual Private Network), etc.
* **Poner en ingles esp-**abreviacio-(ingles)

# Capítulo 4

# Solución Propuesta

En este capítulo se describe la metodología seleccionada para la solución, como así también las herramientas utilizadas, y además los modelos.…

## Materiales y Métodos

Para el desarrollo del presente trabajo se utilizó la metodología CRISP-DM[20] como guía, ya que hoy en día es un estándar de facto para los proyectos de ciencia de datos y minado de datos[21].

Las fases generales de la metodología CRISP-DM son las siguientes:

* Comprensión del negocio:

En esta fase inicial se comprenden los objetivos y los requisitos para cumplir el proyecto de Pronóstico de Series de Tiempo de Tráfico web, donde también se realizaron entrevistas a los trabajadores del área de Departamento de Gestión de Recursos de Redes y Comunicaciones.

Esta fase inicial se enfoca en comprender los objetivos y requisitos del proyecto desde una perspectiva de negocios, y luego convertir este conocimiento en una definición del problema de minería de datos y un plan preliminar del proyecto diseñado para lograr los objetivos[22].

* Comprensión de los datos:

En esta fase se comenzó a ver como extraer los datos de *Google analytics*, en que formato se pueden descargar los datos de la nube, de los datos que están en la nube sobre tráfico web correspondientes a las páginas de la universidad entender como son recolectados esos datos y que significa cada uno.

Agregar un ejemplo ….

Esta fase inicia con recolectar los datos y continúa con actividades para familiarizarse con los datos, identificar problemas de calidad de los datos, descubrir los primeros conocimientos de los datos o detectar subconjuntos interesantes para formular hipótesis[22].

* Preparación de los datos:

En esta fase se procedió a descargar los datos de *Google analytics,* esto a través de la herramienta de Looker Studio, también se limpiaron los datos nulos, y se hizo una eliminación de los datos anómalos del *dataset*.

La fase de preparación cubre todas las actividades para construir el dataset final a partir de los datos iniciales. Las tareas de preparación de datos se realizan varias veces y no en ningún orden prescrito[22].

* Modelado de los datos:

En esta fase se seleccionó la arquitectura *encoder/decoder* y la red neuronal GRU a utilizar, como así también se probaron distintos algoritmos que afinaron los modelos a valores óptimos.

En esta fase, se seleccionan y aplican varias técnicas de modelado, y sus parámetros se calibran a valores óptimos[22].

* Evaluación:

En esta fase se diseñaron y realizaron distintas pruebas a los modelos, de ser necesario se volvía a la fase de modelado a ejecutar nuevamente los algoritmos de calibración de los modelos, finalmente los modelos obtenidos con los datos de UA eran probados y reentrenados para ser usados con los datos de GA4.

En la fase de evaluación, los resultados son evaluados con los objetivos comerciales que se definieron. Por lo tanto, los resultados deben interpretarse y deben definirse acciones adicionales[22][21].

* Despliegue:

Los últimos modelos obtenidos son funcionales con los datos de GA4, es decir, se tienen modelos que funcionan con los datos del entorno real de más actual, dado que los datos de UA ya no serán recolectados por *Google analytics*.

En esta última fase se realizan las conclusiones del proyecto de Pronóstico de Series de Tiempo de Tráfico web y se analizan las futuras líneas de investigación.

La creación del modelo generalmente no es el final del proyecto. Según los requisitos, la fase de desarrollo puede ser tan simple como generar un informe o tan compleja como implementar un proceso de minado de datos repetible

## Herramientas

### Python

Python según dice en su página oficial es un lenguaje de programación interpretado, soporta múltiples paradigmas como pudiera ser programación funcional o programación orientada a objetos, Python es un lenguaje multipropósito, es decir, sirve para solucionar distintos tipos de problemas[23], para este trabajo se utilizaron las versiones de Python 3.7.x y 3.10.x.

### Google Drive

Google ofrece una plataforma que permite almacenar archivos para uso personal a sus usuarios en la nube, este servicio además permite compartir archivos entre usuarios de Google Drive, como así también la edición de archivos en tiempo real con las aplicaciones que nos permite conectar[24].

### Universal Analytics (UA)

Esta es una propiedad de Analytics, básicamente según el soporte de Google cuando se registran vistas u otro tipo de datos del trafico web se recopilan en la propiedad; Universal Analytics es la generación anterior de Analytics desde el 1 Julio del 2023 dejó de procesar datos[25].

### Google Analytics 4 (GA4)

GA4 es una propiedad de Analytics, según el soporte de Google, GA4 recoge datos y los procesa de las páginas web y aplicaciones, mientras que UA no va a funcionar más GA4 es la nueva generación, GA4 por su parte recompila datos desde antes de la fecha en la que UA dejo de funcionar[26].

### Looker Studio

Según la documentación de Google es una herramienta para realizar informes y visualizar los datos, esta herramienta permite la conexión de fuente de datos propios de los usuarios, como pudiera ser *Google Analytics*[26], también se pueden exportar datos de tablas de informes de Looker Studio en archivos CSV.

### Microsoft Word

Microsoft Word un procesador de texto, este software es de Microsoft, sirve para crear, editar, leer, escribir documentos de texto[27].

### Microsoft Excel

Microsoft Excel un software para poder crear, leer y editar planillas de cálculos, este software es de Microsoft[27].

### Numpy

Según su documentación oficial es una librería*,* es un estándar para trabajar con datos numéricos en Python, la librería posee estructuras de datos como Arrays multidimensionales, además posee funciones para realizar cálculos matemáticos con dichos Arrays [28].

### Tensorflow

Es una biblioteca según su documentación oficial es de código abierto para el aprendizaje automático[29].

### Keras

Es una biblioteca que sirve de Interfaz de Programación de Aplicaciones-API  (*Aplication Programing Interface*) de alto nivel de Tensorflow, actualmente Keras se encuentra dentro de la biblioteca Tensorflow y es una API de alto nivel oficial de Tensorflow[29].

### Pandas

Es una librería de Python para el análisis y manipulación de datos, posee el objeto dataframe, también funciones para manipular datos del dataframe y funciones para leer y escribir archivos CSV (valores separados por coma)[30].

### Matplotlib

Matplotlib según su página oficial es una librería para la creación de gráficas, animaciones, distintas formas de visualización de datos en Python[31].

### Google Colaboratory

Según su página oficial Colaboratory es un producto de Google Research que permite a cualquier usuario escribir y ejecutar código de Python en la nube. Este producto se pude conectar con Google Drive, para de esta manera tener acceso y poder leer y escribir los archivos de Google Drive[32].

### Git

Git es un sistema de control de versiones, sirve para controlar las versiones de un proyecto, un proyecto es un repositorio de Git, un proyecto comprende todo el conjunto de carpetas y archivos al que se le realiza el control de versiones, de manera que se guardan los cambios y también se pueden recuperar las versiones anteriores del proyecto[33].

### Hyperparametros (*Hyperparameters*)

Son las variables que se utilizaron durante el entrenamiento del modelo, como así también pueden definir la topología del modelo, estas variables tienen un impacto en el rendimiento del modelo[34].

### Época (*Epoch)*

Según la documentación oficial de *Keras* una época *(Epoch)*, es cuando el mdoelo pasa por todo el conjunto de datos[35], en *Keras*, se puede agregar *callbacks* al final de una época para guardar o imprimir información, como por ejemplo guardar el modelo, de forma que si se ejecutan x cantidad de épocas, se pueda recuperar el modelo con mejor rendimiento de la mejor época.

### Trial

Según su documentación oficial, un trial es una prueba, donde cada trial tiene x cantidad de épocas, en un trial se evalúa un conjunto de valores de hyperparámetros, es decir, en cada trial se evalúa una configuración diferente del modelo[36].

### Unidad Recurrente con Puerta – GRU (*Gate Recurrent Unit*)



Figura 2: Red Neuronal GRU (2).

Donde r es la compuerta de reinicio (*reset gate)* y z es la compuerta de actualización (*update gate*) y h’ es el candidato a activación y h es la activación[11].

La siguiente formula xxx (ref) es para cada j-esimo valor de la compuerta de reset.

(4)

Donde es la función sigmoidea, W y U son matrices que aprenden, es el estado oculto anterior y x es la entrada

La siguiente formula es para cada j-esimo valor de la compuerta de actualización.

(5)

Donde es la función sigmoidea W y U son matrices que aprenden, es el estado oculto anterior y x es la entrada

La siguiente formula es para cada j-esimo valor de activación, o también llamado estado oculto.

(6)

Donde es el candidato a activación y se define en la siguiente fórmula para cada j-esimo valor.

(7)

Donde es la función de tangente hiperbólica y es el producto de Hadamard, W y U son matrices que aprenden, es el estado oculto anterior y x es la entrada[11].

### Arquitectura Codificador-Decodificador (*Encoder-Decoder*)

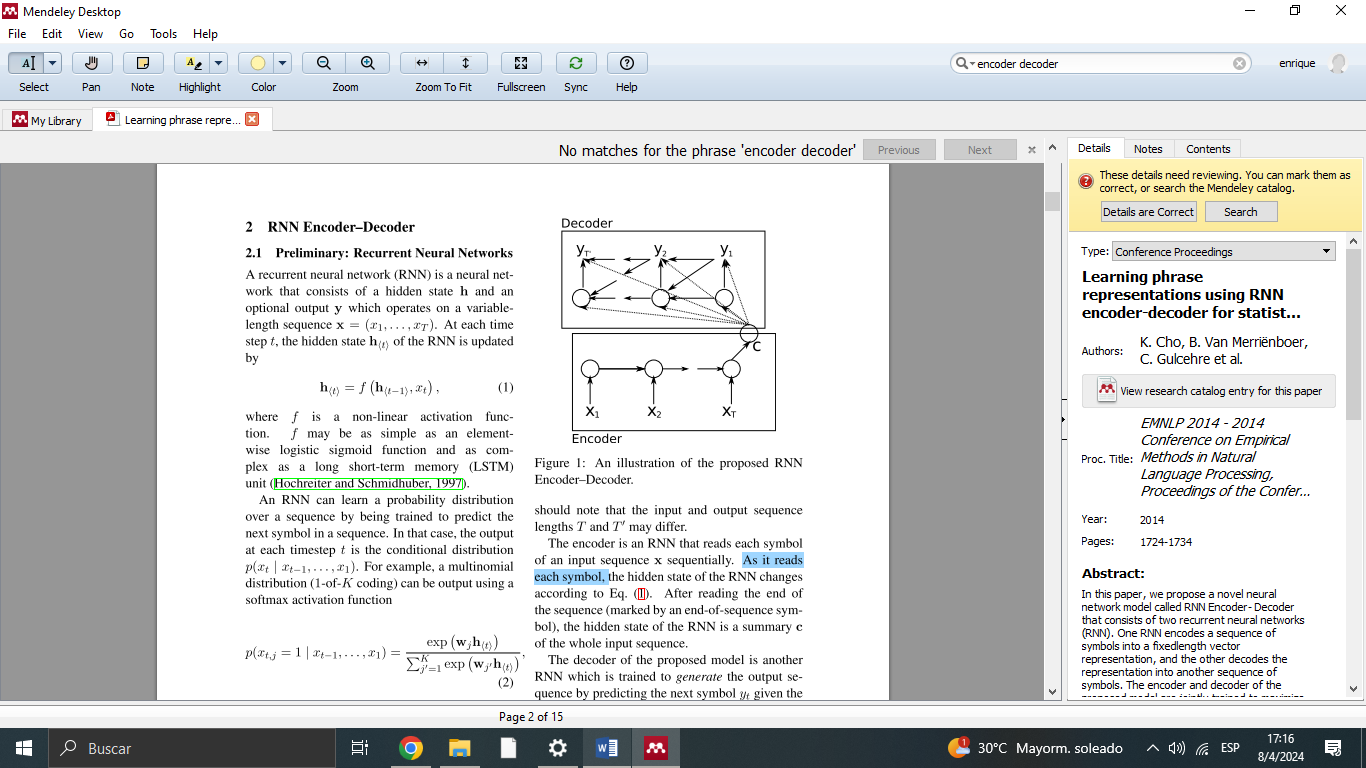


Figura 2: Arquitectura *encoder-decoder* GRU (3).

La arquitectura Codificador-Decodificador consta de un codificador, siendo el codificador una capa de GRU que lee los valores de la secuencia de entrada en orden, y modifica su estado oculto con cada nuevo valor, por otro lado el decodificador también es una capa de GRU que toma como entrada un vector c de los estados ocultos del codificador, de esta manera el decodificador tendría que aprender a predecir el siguiente valor de la secuencia[11].

### Algoritmo Optimización Bayesiana (*Bayesian Otimization*)

La optimización bayesiana es un algoritmo que se utiliza para el afinamiento de los hyperparametros, este algoritmo crea un modelo probabilístico, para la métrica, dado el conjunto de hyperparametros a optimizar [37].

### Algoritmo Hyperband

Es un algoritmo que se usa para el afinamiento de los hyperparametros, este algoritmo ejecuta el algoritmo de *Successive Halving*[38]variando como entrada n y r, siendo n el número de configuraciones y r el número de recursos, como ser numero de epocas, para de esta manera encontrar la mejor configuración de hyperparametros[38], a diferencia de la búsqueda aleatoria y la Optimizacion bayesiana el algoritmo Hyperband no tiene como dato de entrada un número máximo de trials, solo los valores de R (épocas) y n el cual por defecto es 3.

### Algoritmo Búsqueda Aleatoria (*Random Search*)

La Busqueda Aleatoria es un algoritmo que se utiliza para el afinamiento de los hiperparametros, este algoritmo crea muestras aleatorias de las configuraciones de los hyperparametros[39].

### Detención Temprana (*Early Stopping)*

La detención temprana es una configuración que sirve para cortar el entrenamiento de manera temprana[40], esto es, se configura de manera que si el rendimiento no mejora en un número x de épocas se detiene el trial. Por ejemplo si se configura un modelo para que entrene 150 épocas pero la detención temprana está configurada en 20, y al entrenar el modelo no mejora el rendimiento desde la época 40 a la 60, este se detiene y no continúa el entrenamiento.

En este documento científico[41] se menciona el uso de una detención temprana de 10, además en este otro documento científico[40] se menciona el uso la detención temprana para evitar el sobre entrenamiento.

### Github

Github según su página oficial es una plataforma que hospeda repositorios de Git, ofreciendo además el servicio de que los equipos de desarrollo puedan trabajar en grupo[33].

## 4.3 Comprensión de los datos

Esta fase inicia con recolectar los datos y continúa con actividades para familiarizarse con los datos, identificar problemas de calidad de los datos, descubrir los primeros conocimientos de los datos[22].

### 4.3.1 Recolección de datos iniciales

Se tienen acceso a datos históricos de las siguientes 3 páginas webs:

* Sitio Institucional de la U.Na.M. disponible en <https://unam.edu.ar/>
* Sitio de la editorial Universitaria disponible en <https://editorial.unam.edu.ar/>
* Portal de acceso al contenido generado por la radio y la televisión de la Universidad disponible en <https://transmedia.unam.edu.ar/>

De las cuales tenemos mayor cantidad de datos de registrados por UA y menor cantidad de datos expresados en días registrados en GA4, las siguientes tablas explican los datos correspondientes por fecha que se tienen de dichas páginas web.

|  |  |  |
| --- | --- | --- |
| Página Web | Desde | Hasta |
| <https://unam.edu.ar/> | 28-05-2018 | 04-07-2023 |
| <https://editorial.unam.edu.ar/> | 08-10-2018 | 04-10-2023 |
| <https://transmedia.unam.edu.ar/> | 22-08-2018 | 01-06-2023 |

Tabla 1 de datos de Universal Analytics

|  |  |  |
| --- | --- | --- |
| Página Web | Desde | Hasta |
| <https://unam.edu.ar/> | 14-06-2022 | 23-11-2023 |
| <https://editorial.unam.edu.ar/> | 14-06-2022 | 24-01-2024 |
| <https://transmedia.unam.edu.ar/> | 14-06-2022 | 01-01-2024 |

Tabla 2 de datos de Google Analytics 4

Para poder descargar los datos se procedió a conectar Google Analytics como fuente de datos de Looker Studio, de esta manera se pueden pasar los datos a tablas en Looker Studio y posteriormente descargar los datos en archivos CSV.

### 4.3.2 Preparación de los datos

Identificando algunos problemas de calidad de los datos la página web <https://unam.edu.ar/> desde noviembre del año pasado no está capturando el tráfico correctamente en GA4, por lo que se procede a descartar este conjunto de datos.

Lo siguiente que se procedió a realizar, comenzando por la página web <https://transmedia.unam.edu.ar/> se descargaron los datos en archivos csv del Looker Studio de UA, son 5 archivos CSV que se procede a unificar en un solo archivo a través de código en Python.

Se descargaron los siguientes datos en 5 archivos CSV:

Vistas por Sistemas Operativos, Vistas por Agrupación de canales predeterminados, número de vistas de página, Usuarios Nuevos, Usuarios, Numero de Sesiones Por usuario, Sesiones, Vistas por Categoría de dispositivo, Vistas por País

Se procedieron a transformar los datos obtenidos de los archivos CSV de Looker Studio para obtener las columnas del dataframe.

De vistas por sistema operativo salieron las siguientes columnas:

Windows, Android,Linux, Macintosh, Sistema operativo (not set), iOS, Windows Phone,Chrome OS, Tizen, Sistema operativo promedio, Sistema operativo std

Siendo Sistema operativo (not set) donde se registran las vistas los sistemas operativos que UA no pudo detectar, Sistema operativo promedio el promedio de las otras columnas de sistemas operativos, Sistema operativo std el desvio estándar del promedio y el resto son vistas de sistemas operativos conocidos, por ejemplo la columna Windows tendrá en cada fila datos de las vistas de los usuarios que accedieron a la página web a través de un dispositivo cuyo sistema operativo es Windows.

De vistas por Agrupación de canales predeterminados salieron las siguientes columnas:

Organic Search, Social, Direct, Referral, Agrupación de canales predeterminada promedio, Agrupación de canales predeterminada std

Donde Organic Search, Social, Direct y Referral son los canales predeterminados, es decir, las fuentes de tráficos más comunes[42], Agrupación de canales predeterminada promedio es el promedio de los canales predeterminados, y Agrupación de canales predeterminada std es el desvío estándar.

De vistas por Categoría de dispositivo salieron las siguientes columnas:

Mobile, Desktop, Tablet, Categoría de dispositivo promedio, Categoría de dispositivo std.

Donde Mobile, Desktop y Tablet son la Categoría de dispositivo, es decir, el tipo de dispositivo que tenía el usuario cuando se registró la visita en la página web[43], además Categoría de dispositivo promedio es el promedio de vistas y Categoría de dispositivo std es el desvió estándar.

Las siguientes son tal cual del CSV no se necesitó mayor trabajo de transformación:

Número de vistas de página, Usuarios Nuevos, Usuarios, Número de Sesiones Por usuario, Sesiones

Es decir una columna de Usuarios Nuevos del CSV se pasó al dataframe como una columna de Usuarios Nuevos sin mayor preprocesado.

De vistas por País se procedió a dividir en 2 columnas, las vistas que son de Argentina y las que son de otros países.

Luego, se procedió a confirmar la existencia de NaN (Not an number), y eliminar las filas que posean NaNs, el siguiente paso en el preprocesado fue agregar la columna mes, la cual número del mes, luego las columnas de correlación anual que es el promedio de vistas por año y trimestral que básicamente son el promedio de vistas por trimestre.

Finalmente se procedió a calcular el predictor Flag (no se como citar esto)

El predictor flag se calcula tomando el min y máximo de los datos, de ese rango lo divido por un número, en este caso se comenzó con 30 y se pasó a 15, esta división me da el rango de vistas por etiqueta, la etiqueta del rango será el valor medio, luego se crea una columna llamada predictor Flag, en la que por cada fila de la columna de total de vistas (las vistas reales medidas por Google Analytics) se le asigna una etiqueta que coincida con un rango previamente calculado.

(8)

Donde RPE es el rango por etiqueta y N es el número de etiquetas, las etiquetas que se utilizaron posteriormente es el valor medio de cada rango; cuando una vista coincide dentro de un rango, se le asigna la etiqueta correspondiente a dicho rango.

Quedando como resultado del preprocesado las siguientes columnas de tum transmedia, que fueron las mismas columnas que se tomaron en cuenta para las otras páginas web de la institución.

Listado de columnas del Dataframe:

Predictor Flag, Número de vistas de página, Usuarios nuevos, Usuarios, Número de sesiones por usuario, Sesiones, Argentina, Otros Paises, Mobile, Desktop, Tablet, Categoría de dispositivo promedio, Categoría de dispositivo std, Windows, Android, Linux, Macintosh, Sistema operativo (not set), iOS, Windows Phone, Chrome OS, Tizen, Sistema operativo promedio, Sistema operativo std, Organic Search, Social, Direct, Referral, Agrupación de canales predeterminada promedio, Agrupación de canales predeterminada std, Mes, correlación anual, correlación Trimestral

Siendo un total de 33 columnas, por supuesto que desde Google Analytics se registran más datos, sin embargo debido al cambio de UA a GA4[44] solo se eligieron los datos que a nivel tanto conceptual como de medición, fueran más parecidos entre sí.

Finalmente se realizó una limpieza de los datos anómalos- *outliers* detectados usando los k vecinos más cercanos - KNN (*K Nearest Neighbors*)(citar), donde se tomaron los 7 vecinos más cercanos.

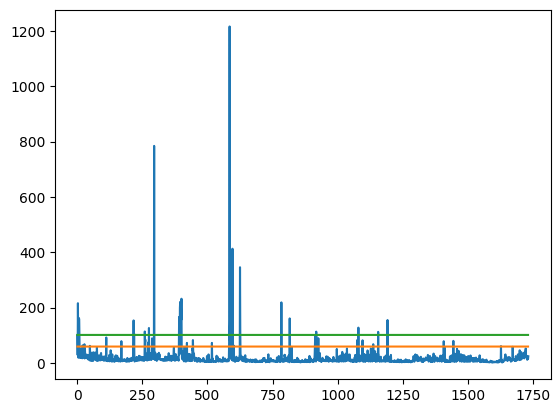


Figura 3: Distancia Promedio de los 7 Vecinos más cercanos para cada fila del dataframe de Tum Transmedia(3).

En la figura anterior muestra la distancia Promedio de los 7 vecinos más cercanos para cada fila, la línea amarilla de valor constante es 59.5981 la media más un desvió estándar, mientras que la línea verde de valor constante es 101.98 la media más dos desvíos estándar, consideramos que si quitamos todos los datos por arriba de 101.98 quitaríamos demasiados, de esta manera podemos considerar los que se encuentren más lejanos de sus 7 vecinos más cercanos como un outlier, en este caso se quitaron las tuplas del conjunto de datos que estén por encima de 500, finalizando así el proceso de limpieza de los datos de tum transmedia.

## **5 Modelos**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tecnologías** | **Red Neuronal** | **Resultados  según Artículos científicos** | **Finalidad** | **Aplica** |
| *Multi-Layer Perceptron*  (MLP) | SI | - | Clasificación, predicción | No |
| *Long-Short Term Memory*  (LSTM) | SI | Combinándolo con otras tecnologías se tiene mejor resultado que usarlo individualmente. | Predictivo, hallar patrones | No |
| ARIMA | No | Buenos | No es redes neuronales, es estadística  predictiva (predicción de series temporales) | No |
| *Gate recurrent unit* (GRU) | SI | El modelo ganador de Kaggle utiliza un cuDNN GRU que es más rápido que un GRU normal. | Sistemas predictivos basados en información secuencial. | No |
| *Bidirectional-LSTM* | SI | Mejor rendimiento comparándolo en el *M3-Competition* | Reconocimiento y clasificación | No |
| GAN | SI | Regular, no hay ganancia comparado con ARIMA | Creación artificial | No |
| RNN seq2seq | SI | Buenos | Modelo ganador del concurso de *kaggle* de *web traffic* 2017 | Si |
| NARX -QPSO | SI | Buenos | Predecir el siguiente valor de la señal de entrada | Si |
| *Prophet-LGBM* | SI | Buenos Mejores resultados que modelos individuales | Combinación ligthGBM y Prophet para predecir trafico web | Si |

Tabla 3 Comparación de tecnologías

## **3.5 Selección de Modelos**

En la Tabla 1 se presenta una comparación de las tecnologías, de ahí se procede a seleccionar las tecnologías que si aplican, son las que se consideran que se podrían usar para la solución del problema; la razón de descartar algunas tecnologías son porque no usan redes neuronales, otras son tecnologías cuya finalidad no es la del pronóstico de series de tiempo y por último, de acuerdo a los artículos científicos presentados en el marco teórico, algunas tecnologías no tienen buen rendimiento o el rendimiento suficiente para ser tomado en cuenta.

Porque el modelo no es lo suficientemente complejo, y tampoco hay tantos datos.

Entre los modelos que quedan seleccionados están RNN seq2seq, NARX –QPSO y Prophet-LGBM de estos modelos Prophet-LGBM si bien se mencionan las fechas de los datos que utilizaron no está disponible el dataset para compararlo con los datos de la problemática de esta tesis por lo tanto Prophet-LGBM se descarta como posible tecnología para la solución de la problemática de esta tesis, mientras que por otro lado tanto RNN seq2seq y NARX –QPSO ambos utilizan el mismo conjunto de datos el cual está disponible online para que cualquiera lo pueda descargar, de esta manera y dado que RNN seq2seq es el ganador del concurso de Kaggle(citar) y por lo tanto se encuentra más información de la misma en la página de Kaggle se opta por utilizar esta tecnología.

En cuanto a la métrica seleccionada para la medición del desempeño de los modelos, elijo la métrica SMAPE, debido a que es la misma con la que se evaluó el modelo RNN seq2seq, que es el ganador del concurso de Kaggle.

# Desarrollo

**Plan de pruebas**

El plan de prueba consiste generar varios modelos GRU con la arquitectura seq2seq, y se procederá a medir usando la métrica SMAPE, en primera instancia se utilizarán los datos de Tum transmedia de UA, posteriormente se utilizará el mejor modelo resultante en los datos de Tum transmedia de GA4, también en los datos de editorial universitaria UA y el mejor modelo resultante de editorial universitaria UA se utilizará para generar un modelo con los datos de editorial universitaria GA4.

(nose si utilizar la palabra reentrenar, generar, entrenar, transferir, entre otros)

**Construcción de los Modelos**

En esta sección se detallan las características de los modelos generados, básicamente en primera instancia se generaron 4 modelos por cada algoritmo de Hypertuner, siendo los algortimos de Hypertuner Hyperband, Bayesian optimisation y el de Random Search. Siendo un total de 12 modelos, tomando datos de 7, 14, 21 y 28 días anteriores para pronosticar los siguientes 7 días, además, se partieron los datos 80% para entrenamiento y 20% para prueba, a la vez, dentro del 80% de datos para el entrenamiento se vuelven partir los datos, por ejemplo, tomando los primeros 28 días para pronosticar los siguientes 7 días, siendo los primeros 28 días la secuencia de entrada y los siguientes 7 días para validación y para ajustar los pesos de la red neuronal, luego a partir del día 2 del conjunto de datos de entrenamiento hasta el día 29 para entrenar y del 30 al 36 para validar, todo ese proceso de partir los datos se vuelve a realizar en el 20% de datos de pruebas.

**Cantidad de Trials por Modelo**

Cada trial tiene 100 epochs y se corrieron un total de 50 trials por modelo. Respecto a la medición, por cada epoch se tiene un rendimiento, entonces por cada trial solo se guarda la epoch con mejor rendimiento, como aclaración el algoritmo hyperband se configuro en 100 épocas pero no tiene el concepto de trials.

**Hyperparametros que se Afinaron**

|  |  |  |
| --- | --- | --- |
| Nombre | Valores posibles | Definición |
| Unidades | 128-512 con un paso de 32 | Es la cantidad de unidades de GRU por capa. |
| Ratio de Aprendizaje | Entre 1e-4 y 1e-2 | Es el ratio de aprendizaje, *Learning Rate* en ingles  Se utiliza un Optimizador Adam. |
| Dropout | 0-0.5 con un paso de 0.1 | El dropout apaga cierto número de neuronas, para evitar el sobre entrenamiento. |
| Recurrent dropout | 0-0.5 con un paso de 0.1 | Apaga un número aleatorio de conexiones recurrentes, para evitar el sobre entrenamiento. |
| Batch size | 32 a 128 con un paso de 32 | Tamaño de lote a tomar, normalmente, esto es mara dividir la cantidad de muestras a tomar, ya que no se puede pasar todo el conjunto de datos entero de una sola vez. |

Tabla 4 Hyperparametros que se afinaron.

Un concepto muy importante que no se podía afinar con la librería keras es el N-Días el cual es básicamente la cantidad de días hacia atrás que toma como datos de entrada la red neuronal, ya que este valor no se puede afinar por la librería keras, se procedió a probar de 7 en 7, ya que este valor modifica el tamaño de las muestras se va variando en cada prueba por cada hyperparametro.

En todos los modelos probados si se toma 28 días anteriores como entrada, se tiene mejor desempeño en todos los algoritmos.

|  |  |
| --- | --- |
| Algoritmo | Desempeño en SMAPE |
| Hyperband | 0.07861421257257462 |
| Bayesian | 0.07861771434545517 |
| Random Search | 0.07861848175525665 |

Tabla 5 Desempeño de algoritmos tomando 28 días anteriores como entrada

Tomando el mejor modelo hyperband obtenido se aumentó la cantidad de paciencia para la parada, y se le reentrenó por mas épocas, siendo 150 épocas, este modelo tendría un desempeño de 0.07860194146633148 SMAPE, dado que la mejora fue ínfima este último se considera el mejor modelo obtenido. También, dado que 28 días era el límite superior respecto a días anteriores de entrada la red neuronal, se probó el algoritmo Hyperband aumentando la cantidad de días que tomaría como entrada en 35 y 42, los modelos resultantes tuvieron un peor desempeño.

Dado que la página web la mayoría de los días caían dentro de la primera etiqueta del predictor flag de 35, esto hace que el modelo pronostique 35 como salida, por lo tanto para confirmar que no sea un error del entrenamiento.

Se realizaron otras pruebas donde se variaron los datos de entrenamiento y prueba respecto al tiempo donde se descartaron partes de los datos que eran solo 35 y se entrenó con partes de los datos donde las vistas eran mayores a 35, también una prueba donde se varió el porcentaje de entrenamiento y prueba en un 70-30.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Algoritmo | N-Días | Unidades | Ratio de Aprendizaje | Dropout | Recurrent Dropout | Batch Size | Paciencia  de Parada | SMAPE |
| Random search | 7 | 288 | 0.001325242185767969 | 0.4 | 0.2 | 128 | 10 | 0.0834403708577156 |
| Random search | 14 | 224 | 0.00036448699004144517 | 0.1 | 0.1 | 32 | 10 | 0.08397094905376434 |
| Random search | 21 | 352 | 0.0018500304177758314 | 0.2 | 0.4 | 32 | 10 | 0.07923400402069092 |
| Random search | 28 | 384 | 0.001458062655405858 | 0.1 | 0.0 | 96 | 10 | 0.07861848175525665 |
| Hyperband | 7 | 128 | 0.007402925076656829 | 0.4 | 0.1 | 128 | 10 | 0.083434097468853 |
| Hyperband | 14 | 448 | 0.0020843011585253736 | 0.3 | 0.4 | 64 | 10 | 0.0839906632900238 |
| Hyperband | 21 | 320 | 0.00020176544783741468 | 0.2 | 0.0 | 32 | 10 | 0.07927556335926056 |
| Hyperband | 28 | 416 | 0.0015323427281788212 | 0.1 | 0.1 | 32 | 10 | 0.07861421257257462 |
| Bayesian | 7 | 416 | 0.0001 | 0.4 | 0.0 | 32 | 10 | 0.08343570679426193 |
| Bayesian | 14 | 224 | 0.0002981287960783473 | 0.1 | 0.4 | 32 | 10 | 0.08396006375551224 |
| Bayesian | 21 | 512 | 0.0008669517422511968 | 0.3 | 0.2 | 32 | 10 | 0.07924449443817139 |
| Bayesian | 28 | 256 | 0.0001 | 0.0 | 0.3 | 32 | 10 | 0.07861771434545517 |

Tabla 6 Pruebas de Algoritmos de Afinamiento de Hyperparametros.

Enumerar cada prueba? Ponerle un nombre a cada prueba? Para referirme a ellas en un futuro?

También se probó el modelo sin pasar por el predictor flag.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Algoritmo | N-Días | Unidades | Ratio de Aprendizaje | Dropout | Recurrent Dropout | Batch Size | Paciencia  de Parada | SMAPE |
| Hyperband | 7 | 480 | 0.0003396472862594332 | 0.1 | 0.4 | 32 | 10 | 0.61905437707901 |
| Hyperband | 14 | 480 | 0.003732448879534993 | 0.2 | 0.2 | 32 | 10 | 0.6086962223052979 |
| Hyperband | 21 | 416 | 0.003401869192259701 | 0.0 | 0.1 | 96 | 10 | 0.6049731969833374 |
| Hyperband | 28 | 352 | 0.0038812620867338375 | 0.1 | 0.2 | 32 | 10 | 0.5994085073471069 |

Tabla 7 Pruebas de Modelod sin pasar por el predictor flag.

Dado que hyperband fue el mejor algoritmo con N-Días como 28 se ejecutó el mejor modelo del mejor algoritmo aumentando el *early stopping* de 10 a 50 y aumentando las épocas a 150.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Algoritmo | N-Días | Unidades | Ratio de Aprendizaje | Dropout | Recurrent Dropout | Batch Size | Paciencia  de Parada | SMAPE |
| Hyperband | 28 | 416 | 0.0015323427281788212 | 0.1 | 0.1 | 32 | 50 | 0.07860194146633148 |

Tabla 8 Prueba del mejor Modelo aumentando las épocas y el *early stopping*.

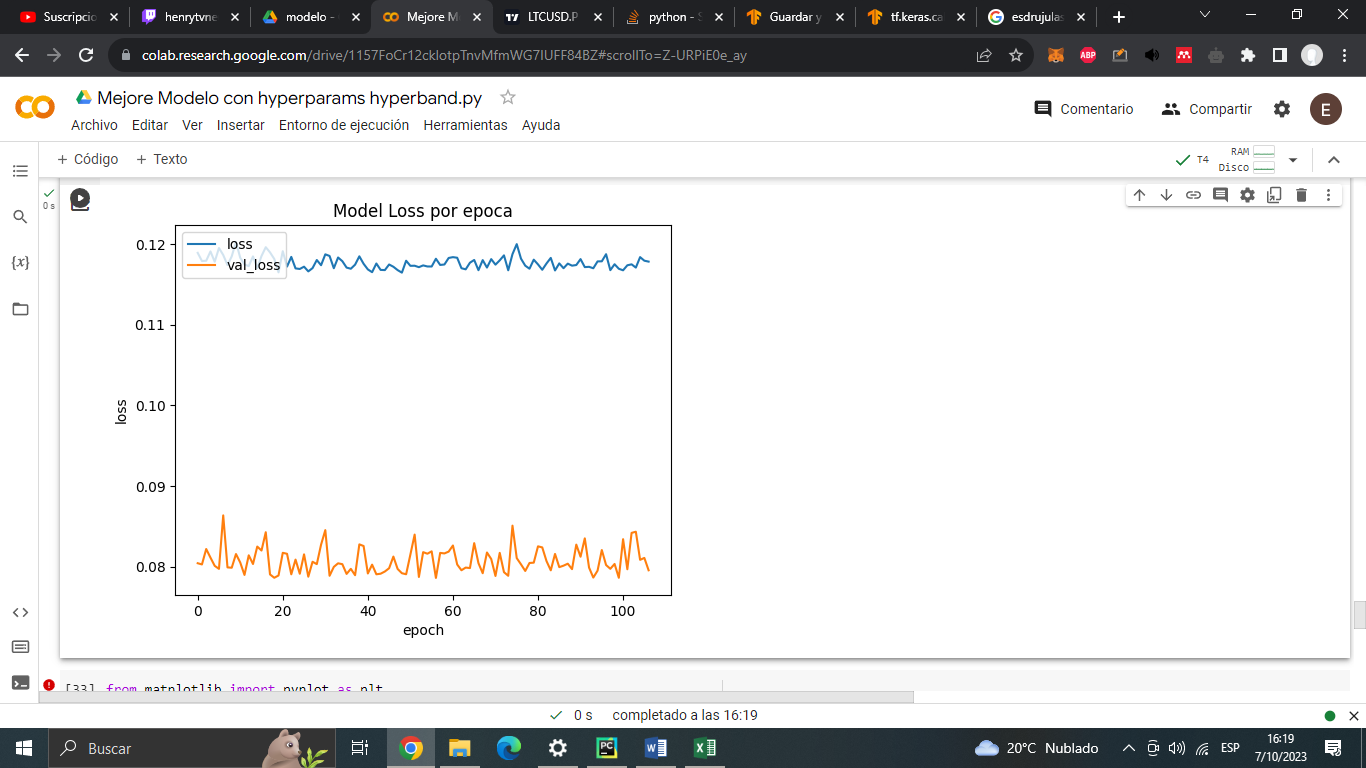


Figura 4: …

En la imagen se puede ver que desde la época 57 no mejora el rendimiento entonces cortó el entrenamiento en la época 107

Dado el mejor modelo se hizo una prueba donde se varió el % que se toma de entrenamiento-prueba.

|  |  |  |
| --- | --- | --- |
| Prueba | Variación % | SMAPE |
| Prueba 1 Variación % | De 20% a 30% Test | 0.06345754861831665 |

Tabla 9 Prueba del mejor Modelo Variando el Porcentaje.

Dado que el Algoritmo Hyperband tuvo mejor rendimiento en N-Days como 28

Y ese fue el límite superior que se probó, también se decidieron hacer pruebas aumentando la variable N-Days.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Algoritmo | N-Días | Unidades | Ratio de Aprendizaje | Dropout | Recurrent Dropout | Batch Size | Paciencia  de Parada | SMAPE |
| Hyperband | 35 | 288 | 0.0001687328693742953 | 0.1 | 0.0. | 32 | 10 | 0.0804145559668541 |
| Hyperband | 42 | 288 | 0.0006199627420337139 | 0.1 | 0.4 | 32 | 10 | 0.08229460567235947 |

Tabla 10 Prueba del algoritmo Hyperband en 35 y 42 N-Días.

También se hicieron pruebas donde se variaron el orden de los datos de entrenamiento-prueba en fecha, esto es, variando el orden de las fechas, donde no necesariamente las fechas del 20% de *test* tiene que ser el último 20% del conjunto de datos con N-Días en 28 y Paciencia de Parada en 10.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Algoritmo | N-Días | Unidades | Ratio de Aprendizaje | Dropout | Recurrent Dropout | Batch Size | Paciencia  de Parada | SMAPE |
| Hyperband | 28 | 288 | 0.0016406586383193105 | 0.1 | 0.4 | 64 | 10 | 0.18816892802715302 |
| Hyperband | 28 | 160 | 0.0007479781936524693 | 0.3 | 0.2 | 32 | 10 | 0.18816962838172913 |
| Hyperband | 28 | 192 | 0.0028156208463983876 | 0.0 | 0.0 | 32 | 10 | 0.1882011443376541 |

Tabla 11 Prueba del algoritmo Hyperband variando fechas.

Más detalle en anexo.

Concluidas las pruebas en tum transmedia se seleccionaron los 2 modelos con mejor rendimiento y se pasaron a probar con los datos la página web de editorial uman.

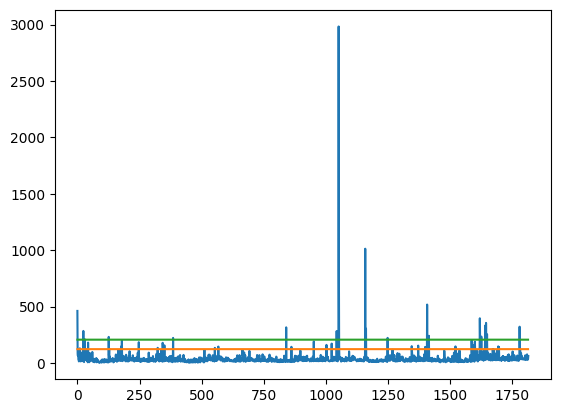


Figura 5: Distancia Promedio de los 7 Vecinos más cercanos para cada fila del dataset de Editorial Universitaria(3).

Se procedió a reentrenar los 2 mejores modelos de Tum Transmedia con los datos de Editorial Universitaria de UA.

|  |  |  |
| --- | --- | --- |
| # | Prueba | Rendimiento |
| 1 | Mejor algoritmo Hyperband 70-30 | 0.29426494240760803 |
| 2 | Mejor algoritmo Hyperband 80-20 | 0.34444767236709595 |

### Conversión de Modelos de UA a GA4 de Tum Transmedia

Se procedió a reentrenar los 2 mejores modelos de Tum Transmedia UA con los datos de Tum Transmedia GA4.

|  |  |  |
| --- | --- | --- |
| # | Prueba | Rendimiento |
| 1 | Mejor algoritmo Hyperband 70-30 | 0.447075 |
| 2 | Mejor algoritmo Hyperband 80-20 | 0.441963 |

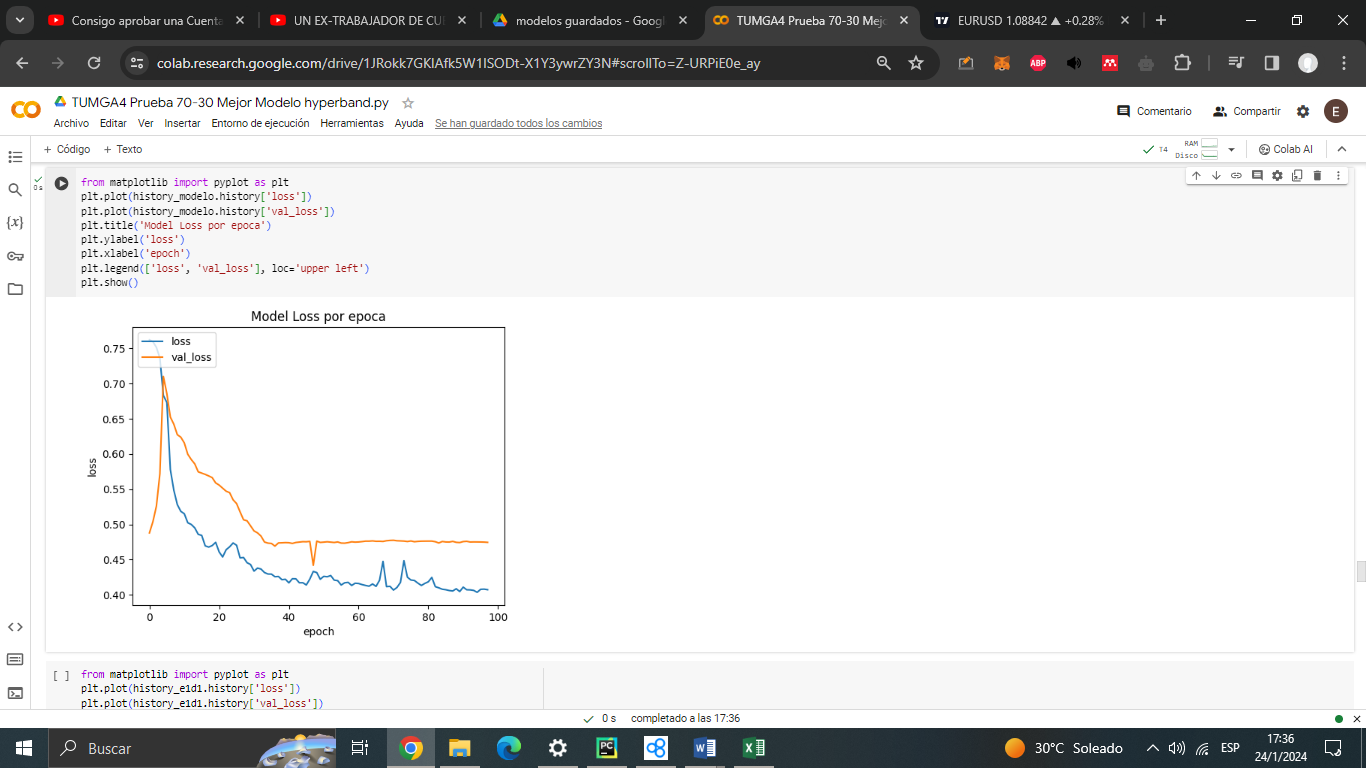


Figura 6: Rendimiento por época del Modelo 70-30 con datos de Tum Transmedia GA4(3).

### Conversión de Modelos de UA a GA4 de Editorial Universitaria

Se procedió a reentrenar los 2 mejores modelos de Editorial Universitaria UA con los datos de Editorial Universitaria GA4.

|  |  |  |
| --- | --- | --- |
| # | Prueba | Rendimiento |
| 1 | Mejor algoritmo Hyperband 70-30 | 0.590801 |
| 2 | Mejor algoritmo Hyperband 80-20 | 0.5983048677444458 |

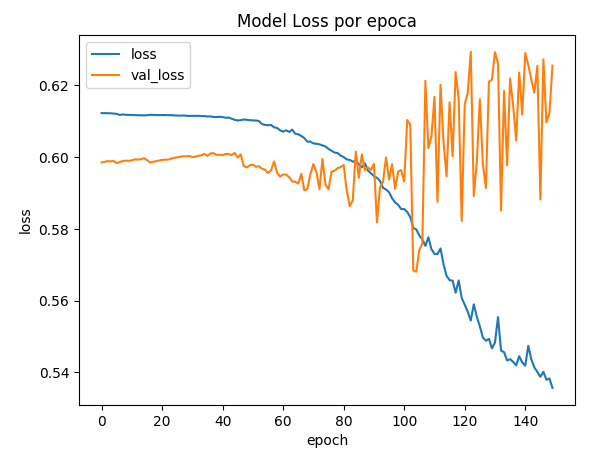


Figura 7: Rendimiento por época del Modelo 80-20 con datos de Editorial Universitaria GA4(3).

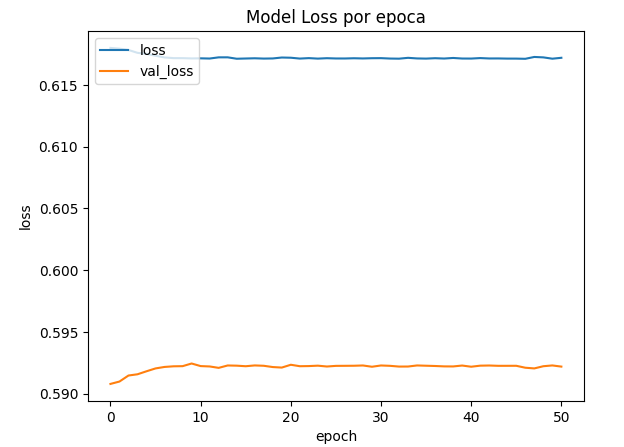


Figura 8: Rendimiento por época del Modelo 80-20 con datos de Editorial Universitaria GA4(3).

To do

Terminar tabla de pruebas

Hacer conclusiones

Hacer Futuras lineas de investigación

Hacer citar lode predictor flag

Poner en alguna parte lo de replicar el modelo

Hacer grafico de knn

Citar a early stopping pacience

Citar Word y excel

Tabla algoritmo

Tambien se pasaron por el algoritmo Hyperband diferentes combinaciones de fechas de manera que no solo se probara la versión en que las fechas en orden en prueba y entrenamiento.

Siendo los modelos de las pruebas del 20 % y 30% los que obtuvieron mejor resultado los que se utilizaron en un futuro.

Preguntas

1. Corregir todos al formato español – acrónimo (nombre en inglés) si-no
2. Deberia explicar mas en detalle los algoritmos de hypertuner? si-no
3. Debería de explicar antes las variables que se afinaron? si-no
4. Dado que para las pruebas utilizo la palabra prueba; mientras que para entrenamiento y prueba también utilizo la palabra prueba debería de cambiar la palabra? si-no
5. Palabras en inglés como hypertuner, hyperparameter, outlier, dataframe, hyperband etc todas deberían de traducirse? si-no

# Capítulo 5

# Conclusión

En este capítulo se pretende….

## 5.1 **Conclusiones**

El desarrollo de este trabajo aporta a la U.Na.M modelos de Pronóstico de Trafico Web; el mejor modelo con los datos de UA tiene un SMAPE de 0.06345754861831665 correspondiente a los datos de Tum Transmedia, mientras que el mejor modelo en el documento científico[1] donde se pronostican páginas web de Wikipedia, tiene un SMAPE de 0.349, dado que menores números de SMAPE se traducen en un mejor resultado, concluimos que es un buen resultado el modelo obtenido para la página web de Tum Transmedia, por otro lado, tenemos el mejor modelo obtenido para la página web de la Editorial Universitaria, este modelo tiene un SMAPE de 0.34444767236709595, un valor bastante similar en el que se menciona se obtuvo en el documento científico[1], también se realizaron pruebas de pasar los modelos con los datos de UA a GA4 dichos modelos tuvieron un peor rendimiento, probablemente debido a que GA4 registra de manera distinta los datos que en UA, si bien se tomaron columnas que en significado son lo mismo, las formas de medir los datos en GA4 cambian, también si vemos los datos, UA toma datos desde 2018 pre-pandemia, y GA4 desde 2022 pos-pandemia, esto también pudo influenciar en los resultados y un modelo que aprendió con datos pre-pandemia tal vez no pudo ni siquiera re-entrenado captar las características nuevas pos-pandemia, dado que los modelos de GA4 reentrenados tuvieron un conjunto reducido de datos, está la posibilidad de que si contáramos con una mayor cantidad de datos, pudiéramos tener un mejor rendimiento en el pronóstico.

## 5.2 Futuras líneas de investigación

A partir del desarrollo de esta Tesis surgen varias líneas de trabajo que se enumeradas a continuación:

* Desarrollar un sistema que realice el pronóstico de series de tiempo de tráfico web en tiempo real.
* Realizar un estudio de caso comparativo de diferentes IAs con los datos de las páginas web de la U.Na.M.
* Generar nuevos modelos agregando nuevos datos que solo se miden en GA4 que no se medían en UA.
* Utilizar los resultados del Pronóstico de Series de Tiempo de Tráfico Web para pronosticar el uso de recursos de hardware.
* Probar los Modelos generados con otras páginas web de la U.Na.M.
* Realizar un estudio estadístico para comprobar que los datos de GA4 son parecidos a los de UA.
* Entrenar modelos de pronóstico de series de tiempo de tráfico web cuando se tengan una mayor cantidad de datos de GA4.

# Anexo

Detalle de pruebas donde se variaron el orden de los datos de entrenamiento prueba en fecha.

* Prueba 1 variando las fechas de entrenamiento y prueba

train\_df,test\_df = dataset[1:950], dataset[950:1250]

train\_df2 = dataset[1250:1670]

test\_df2 = dataset[1670:]

Hyperparameters:

Unidades: 288

Ratio de Aprendizaje: 0.0016406586383193105

*Dropout*: 0.1

*Recurrent Dropout*: 0.4

*Batch size*: 64

*Score*: 0.18816892802715302

* Prueba 2 variando las fechas de entrenamiento y prueba

train\_df,test\_df = dataset[1:950], dataset[950:1250]

train\_df2 = dataset[1250:1670]

test\_df2 = dataset[1670:]

Misma configuración de datos train-test que el anterior pero con más épocas.

Hyperparametros:

Unidades: 160

Ratio de Aprendizaje: 0.0007479781936524693

*Dropout*: 0.30000000000000004

*Recurrent* *Dropout*: 0.2

*Batch size*: 32

Epocas:200

*Score*: 0.18816962838172913

* Prueba 3 variando las fechas de entrenamiento y prueba

train\_df,test\_df = dataset[0:100], dataset[950:1250]

train\_df2 = dataset[250:500]

train\_df3 = dataset[1360:1490]#dataset[1250:1670]

test\_df2 = dataset[1670:]

Hyperparametros:

Unidades: 192

Ratio de Aprendizaje: 0.0028156208463983876

*Dropout*: 0.0

*Recurrent Dropout*: 0.0

*Batch size*: 32

*Score*: 0.1882011443376541

Pruebas con datos de GA4

Agregar anexo de preprocesamiento de datos de UA a GA4

Agregar knn de Ga4 tum y editorial

Agregar knn de 5 en vez de 7 para mostrar que se probó con diferentes valores

# Bibliografía

[1] N. Petluri and E. Al-Masri, “Web Traffic Prediction of Wikipedia Pages,” *Proc. - 2018 IEEE Int. Conf. Big Data, Big Data 2018*, pp. 5427–5429, 2019.

[2] D. Quoc Nguyen, M. Nguyet Phan, and I. Zelinka, “Periodic Time Series Forecasting with Bidirectional Long Short-Term Memory: Periodic Time Series Forecasting with Bidirectional LSTM,” *ACM Int. Conf. Proceeding Ser.*, pp. 60–64, 2021.

[3] V. Kotu and B. Deshpande, “Chapter 12 - Time Series Forecasting,” V. Kotu and B. B. T.-D. S. (Second E. Deshpande, Eds. Morgan Kaufmann, 2019, pp. 395–445.

[4] K. Zhou, W. Wang, L. Huang, and B. Liu, “Comparative study on the time series forecasting of web traffic based on statistical model and Generative Adversarial model,” *Knowledge-Based Syst.*, vol. 213, p. 106467, Feb. 2021.

[5] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[6] K. Zhou, W. Wang, L. Huang, and B. Liu, “Comparative study on the time series forecasting of web traffic based on statistical model and Generative Adversarial model,” *Knowledge-Based Syst.*, vol. 213, Feb. 2021.

[7] “Prophet: forecasting at scale.” [Online]. Available: https://research.facebook.com/blog/2017/2/prophet-forecasting-at-scale/. [Accessed: 24-Oct-2020].

[8] R. Casado-Vara, A. M. del Rey, D. Pérez-Palau, L. De-La-fuente-valentín, and J. M. Corchado, “Web Traffic Time Series Forecasting Using LSTM Neural Networks with Distributed Asynchronous Training,” *Math. 2021, Vol. 9, Page 421*, vol. 9, no. 4, p. 421, Feb. 2021.

[9] W. Wang and Y. Lu, “Analysis of the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE) in Assessing Rounding Model,” in *IOP Conference Series: Materials Science and Engineering*, 2018, vol. 324, no. 1.

[10] G. P. Meyer, “An Alternative Probabilistic Interpretation of the Huber Loss,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5257–5265.

[11] K. Cho *et al.*, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2014, pp. 1724–1734.

[12] “Web Traffic Time Series Forecasting,” 2017. [Online]. Available: https://www.kaggle.com/c/web-traffic-time-series-forecasting/discussion/39367. [Accessed: 12-Oct-2022].

[13] “Choosing the correct error metric: MAPE vs. sMAPE,” 2020. [Online]. Available: https://towardsdatascience.com/choosing-the-correct-error-metric-mape-vs-smape-5328dec53fac. [Accessed: 12-Oct-2022].

[14] C. Kuranga and N. Pillay, “A comparative study of nonlinear regression and autoregressive techniques in hybrid with particle swarm optimization for time-series forecasting,” *Expert Syst. Appl.*, vol. 190, Mar. 2022.

[15] P. Khanarsa, A. Luangsodsai, K. Sinapiromsaran, I. F. Astachova, K. A. Makoviy, and Y. V Khitskova, “Possibilities for predicting the state of usability web resources,” *J. Phys. Conf. Ser.*, vol. 1902, no. 1, p. 012029, May 2021.

[16] D. Deng, F. Karl, F. Hutter, B. Bischl, and M. Lindauer, “Efficient Automated Deep Learning for Time Series Forecasting,” 2022.

[17] S. Xu, C. Han, and C. Ran, “A Time Series Combined Forecasting Model Based on Prophet-LGBM,” *ACM Int. Conf. Proceeding Ser.*, May 2021.

[18] “The M3-Competition Database.” [Online]. Available: https://forecasters.org/resources/time-series-data/m3-competition/. [Accessed: 18-Oct-2022].

[19] A. Botchkarev, “Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology,” 2018.

[20] P. Chapman *et al.*, “CRISP-DM 1.0: Step-by-step data mining guide,” *SPSS inc*, vol. 9, no. 13, pp. 1–73, 2000.

[21] C. Schröer, F. Kruse, and J. M. Gómez, “A systematic literature review on applying CRISP-DM process model,” *Procedia Comput. Sci.*, vol. 181, pp. 526–534, 2021.

[22] W. Rüdiger and J. Hipp, “CRISP-DM : Towards a Standard Process Model for Data Mining,” *Proc. Fourth Int. Conf. Pract. Appl. Knowl. Discov. Data Min.*, no. 24959, pp. 29–39, 2000.

[23] Python Software Foundation, “Python 3.12.1 documentation.” [Online]. Available: https://docs.python.org/3/. [Accessed: 02-Feb-2024].

[24] Google, “Plataforma de archivos compartidos y almacenamiento personal en la nube - Google.” [Online]. Available: https://www.google.com/intl/es-419\_ar/drive/. [Accessed: 02-Feb-2024].

[25] Google, “Ayuda de Google.” [Online]. Available: https://support.google.com/. [Accessed: 02-Feb-2024].

[26] Google Lcc, “Ayuda de Google,” *Ayuda de Google*, 2023. [Online]. Available: https://support.google.com/.

[27] “Microsoft: Nube, aplicaciones y juegos.” [Online]. Available: https://www.microsoft.com/es-ar/. [Accessed: 16-Apr-2024].

[28] Numpy.org, “NumPy Documentation.” [Online]. Available: https://numpy.org/doc/. [Accessed: 05-Feb-2024].

[29] Google Lcc, “TensorFlow es una plataforma de código abierto de extremo a extremo para el aprendizaje automático.” [Online]. Available: https://www.tensorflow.org/. [Accessed: 05-Feb-2024].

[30] “About pandas.” [Online]. Available: https://pandas.pydata.org/about/. [Accessed: 06-Feb-2024].

[31] “https://matplotlib.org/.” [Online]. Available: https://matplotlib.org/. [Accessed: 06-Feb-2024].

[32] Google, “Google Colab.” [Online]. Available: https://research.google.com/colaboratory. [Accessed: 02-Feb-2024].

[33] GitHub, “Acerca de Git.” [Online]. Available: https://docs.github.com/es/get-started/using-git/about-git. [Accessed: 06-Feb-2024].

[34] “Introduction to the Keras Tuner.” [Online]. Available: https://www.tensorflow.org/tutorials/keras/keras\_tuner. [Accessed: 15-Feb-2024].

[35] “Keras FAQ.” [Online]. Available: https://keras.io/getting\_started/faq/#. [Accessed: 15-Feb-2024].

[36] “The base Tuner class.” [Online]. Available: https://keras.io/api/keras\_tuner/tuners/base\_tuner/. [Accessed: 15-Feb-2024].

[37] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian Optimization of Machine Learning Algorithms,” *Adv. Neural Inf. Process. Syst.*, vol. 25, 2012.

[38] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: A novel bandit-based approach to hyperparameter optimization,” *J. Mach. Learn. Res.*, vol. 18, pp. 1–52, 2018.

[39] J. Bergstra and B. Yoshua, “Random search for hyper-parameter optimization,” *J. Mach. Learn. Res.*, vol. 13, no. 10, pp. 281–305, 2012.

[40] A. Baghbanpourasl, E. Lughofer, P. Meyer-Heye, H. Zorrer, and C. Eitzinger, “Virtual quality control using bidirectional lstm networks and gradient boosting,” in *IEEE International Conference on Industrial Informatics (INDIN)*, 2019, vol. 2019-July, pp. 1638–1643.

[41] J. Tourille, O. Ferret, X. Tannier, and A. L. Nvol, “Neural architecture for temporal relation extraction: A Bi-LSTM approach for detecting narrative containers,” in *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 2017, vol. 2, pp. 224–230.

[42] “[UA] Información sobre las agrupaciones de canales.” [Online]. Available: https://support.google.com/analytics/answer/6010097. [Accessed: 07-Feb-2024].

[43] “[GA4] Informe ‘Detalles de la tecnología.’” [Online]. Available: https://support.google.com/analytics/answer/12980150. [Accessed: 07-Feb-2023].

[44] “[UA→GA4] Diferencias entre los datos de Universal Analytics y Google Analytics 4.” [Online]. Available: https://support.google.com/analytics/answer/9964640. [Accessed: 10-Feb-2024].